Execution Time of λ-Terms via Denotational Semantics and Intersection Types

Daniel de Carvalho

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Execution Time of λ-Terms via Denotational Semantics and Intersection Types

Daniel de Carvalho

N° 6638

septembre 2008

Thème SYM

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--6638--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
NANCY - GRAND EST

Execution Time of λ-Terms via Denotational Semantics and Intersection Types

Daniel de Carvalho*

Thème SYM — Systèmes symboliques
Équipe-Projet CALLIGRAMME

Rapport de recherche n° 6638 — septembre 2008 — 39 pages

Abstract: The multiset based relational semantics for linear logic induces a semantics for the type free λ-calculus. This one is built on non-idempotent intersection types. We prove that the size of the derivations and the size of the types are closely related to the execution time of λ-terms in a particular environment machine, Krivine's machine.

Key-words: λ-calculus, denotational semantics, intersection types, computational complexity.

This is an edited version of [de Carvalho 2006].

* INRIA-Lorraine, Nancy – Daniel.decarvalho@loria.fr

Centre de recherche INRIA Nancy – Grand Est
LORIA, Technopôle de Nancy-Brabois, Campus scientifique,
615, rue du Jardin Botanique, BP 101, 54602 Villers-Lès-Nancy
Téléphone : +33 3 83 59 30 00 — Télécopie : +33 3 83 27 83 19

Le temps d'exécution des λ-termes via la sémantique dénotationnelle et les types avec intersection

Résumé : La sémantique relationnelle multi-ensembliste de la logique linéaire induit une sémantique du λ-calcul non typé. Celle-ci est construite sur des types avec une intersection non-idempotente. Nous prouvons que la taille des dérivations et la taille des types sont étroitement liées au temps d'exécution des λ-termes dans une machine à environnement particulière, la machine de Krivine.

Mots-clés : λ-calcul, sémantique dénotationnelle, types avec intersection, complexité du calcul.

Introduction

This paper presents a work whose aim is to obtain information on execution time of λ-terms by semantic means.

Execution time means the number of steps in a computational model. As in [Ehrhard and Regnier 2006], the computational model considered in this paper will be Krivine's machine, a more realistic model than β-reduction. Indeed, Krivine's machine implements (weak) head linear reduction: in one step, we can do at most one substitution. In this paper, we consider two variants of this machine : the first one (Definition 4) computes the head-normal form of any λ-term (if it exists) and the second one (Definition 11) computes the normal form of any λ-term (if it exists).

The fundamental idea of denotational semantics is that propositions should be interpreted as the objects of a category $\mathbb{C}$ and proofs should be interpreted as morphisms in $\mathbb{C}$ in such a way that if a proof Π reduces to a proof Π' by cut-elimination, then they are interpreted by the same morphism. By the Curry-Howard isomorphism, a simply typed λ-term is a proof in intuitionistic logic. Now, the intuitionistic fragment of linear logic [Girard 1987] is a refinement of intuitionistic logic. This means that when we have a categorical structure $(\mathbb{C}, \ldots)$ to interpret intuitionistic linear logic, one can derive a category $\mathbb{K}$ that is a model of intuitionistic logic.

Linear logic has various denotational semantics; one of these is the multiset based relational semantics in the category **Rel** of sets and relations with the comonad associated to the finite multisets functor (see [Tortora de Falco 2000] for interpretations of proof-nets and Appendix of [Bucciarelli and Ehrhard 2001] for interpretations of derivations of sequent calculus). Here, the category $\mathbb{K}$ is a category equivalent to the Kleisli category of this comonad. The semantics we obtain is *non-uniform* in the following sense : the interpretation of a function contains information about its behaviour on chimerical arguments (see Example 18 for an illustration of this fact). As we want to consider type free λ-calculus, we will consider λ-algebras in $\mathbb{K}$. We will put semantics of λ-terms in these λ-algebras in a logical framework, using intersection types.

The intersection types system that we consider (System R, defined in Subsection 3.1) is a reformulation of that of [Coppo *et al.* 1980]; in particular, it lacks idempotency, as System λ in [Kfoury 2000] and System $\mathbb{I}$ in [Neergaard and Mairson 2004] and contrary to System $\mathcal{I}$ of [Kfoury *et al.* 1999]. So, we stress the fact that the semantics of [Coppo *et al.* 1980] can be reconstructed in a natural way from the finite multisets relational model of linear logic using the Kleisli construction.

Now, if v and u are two closed normal λ-terms, we can wonder

1. Is it the case that the λ-term $(v)u$ is (head) normalizable?

2. If the answer to the previous question is positive, what is the number of steps leading to the (principal head) normal form?

The main point of the paper is to show that it is possible to answer both questions by only referring to the semantics $[\![v]\!]$ and $[\![u]\!]$ of v and u respectively. The answer to the first question is given in Section 4 (Corollary 34) and is a simple adaptation of well-known results. The answer to the second question is given in Section 5.

The paper [Ronchi Della Rocca 1988] presented a procedure that computes a normal form of any λ-term (if it exists) by finding its principal typing (if it exists). In Section 5, we present some quantitative results about the relation between the types and this computation. In particular, we prove that the number of steps of execution of a λ-term in the first machine is the size of the least derivation of the λ-term in System R (Theorem

44) and prove a similar result for the second machine (Theorem 50). We end by proving truly semantic measures of execution time in Subsection 5.4 and Subsection 5.5.

Notation. We denote by Λ the set of λ-terms, by $\mathcal{V}$ the set of variables and, for any λ-term t, by $FV(t)$ the set of free variables in t.

We use Krivine's notation for λ-terms i.e. λ-term v applied to u is noted $(v)u$.

We use the notation $[\,]$ for multisets while the notation $\{\,\}$ is, as usual, for sets. The pairwise union of multisets given by term-by-term addition of multiplicities is denoted by a $+$ sign and, following this notation, the generalized union is denoted by a $\sum$ sign. The neutral element for this operation, the empty multiset, is denoted by $[]$.

1 Krivine's machine

We introduce two variants of a machine presented in [Krivine 2007] that implements call-by-name. More precisely, the original machine performs weak head linear reduction, whereas the machine presented in Subsection 1.2 performs head linear reduction. Subsection 1.3 slightly modifies the latter machine as to compute the β-normal form of any normalizable term.

1.1 Execution of States

We begin with the definitions of the set $\mathcal{E}$ of environments and of the set $\mathcal{C}$ of closures.

Set $\mathcal{E} = \bigcup_{p \in \mathbb{N}} \mathcal{E}_p$ and set $\mathcal{C} = \bigcup_{p \in \mathbb{N}} \mathcal{C}_p$, where $\mathcal{E}_p$ and $\mathcal{C}_p$ are defined by induction on p:

- If $p = 0$, then $\mathcal{E}_p = \{\emptyset\}$ and $\mathcal{C}_p = \Lambda \times \{\emptyset\}$.
- $\mathcal{E}_{p+1}$ is the set of partial maps $\mathcal{V} \to \mathcal{C}_p$, whose domain is finite, and $\mathcal{C}_{p+1} = \Lambda \times \mathcal{E}_{p+1}$.

For $e \in \mathcal{E}$, $d(e)$ denotes the least integer p such that $e \in \mathcal{E}_p$.
For $c = (t, e) \in \mathcal{C}$, we define, by induction on $d(e)$, $\overline{c} = t[e] \in \Lambda$:

- If $d(e) = 0$, then $t[e] = t$.
- Assume $t[e]$ defined for $d(e) = d$. If $d(e) = d + 1$, then $t[e] = t[\overline{c_1}/x_1, \ldots, \overline{c_m}/x_m]$, with $\{x_1, \ldots, x_m\} = \mathsf{dom}(e)$ and, for $1 \leq j \leq m$, $e(x_j) = c_j$.

A *stack* is a finite sequence of closures. If c is a closure and $\pi = (c_1, \ldots, c_q)$ is a stack, then $c.\pi$ will denote the stack $(c, c_1, \ldots, c_q)$. We will denote by ϵ the empty stack.

A *state* is a pair (c, π), where c is a closure and π is a stack. If $s = (c_0, (c_1, \ldots, c_q))$ is a stack, then $\overline{s}$ will denote the λ-term $(\overline{c_0})\overline{c_1} \ldots \overline{c_q}$.

Definition 1 *We say that a λ-term t* respects the variable convention *if any variable is bound at most one time in t.*

For any closure $c = (t, e)$, we define, by induction on $d(e)$, what it means for c to respect the variable convention*:*

- *if $d(e) = 0$, then we say that c respects the variable convention if, and only if, t respects the variable convention ;*
- *if $c = (t, \{(x_1, c_1), \ldots, (x_m, c_m)\})$ with $m \neq 0$, then we say that c respects the variable convention if, and only if,*

– $c_1, \ldots, c_m$ *respect the variable convention ;*

– *and the variables* $x_1, \ldots, x_m$ *are not bound in* t.

For any state $s = (c_0, (c_1, \ldots, c_q))$, *we say that* s respects the variable convention *if, and only if,* $c_0, \ldots, c_q$ *respect the variable convention.*

We denote by $\mathbb{S}$ *the set of the states that respect the variable convention.*

First, we present the execution of a state (that respects the variable convention). It consists in updating a closure (t, e) and the stack. If t is an application $(v)u$, then we push the closure (u, e) on the top of the stack and the current closure is now (v, e). If t is an abstraction, then a closure is popped and a new environment is created. If t is a variable, then the current closure is now the value of the variable of the environment. The partial map $s \succ_{\mathbb{S}} s'$ (defined below) defines formally the transition from a state to another state.

Definition 2 *We define a partial map from* $\mathbb{S}$ *to* $\mathbb{S}$: *for any* $s, s' \in \mathbb{S}$, *the notation* $s \succ_{\mathbb{S}} s'$ *will mean that the map assigns* s' *to* s. *The value of the map at* s *is defined as follows:*

$$s \mapsto \begin{cases} (e(x), \pi) & \text{if } s = ((x, e), \pi) \text{ with } x \in \mathsf{dom}(e) \\ \text{not defined} & \text{if } s = ((x, e), \pi) \text{ with } x \in \mathcal{V} \text{ and } x \notin \mathsf{dom}(e) \\ ((u, \{(x, c)\} \cup e), \pi') & \text{if } s = ((\lambda x.u, e), c.\pi') \\ \text{not defined} & \text{if } s = ((\lambda x.u, e), \epsilon) \\ ((v, e), (u, e).\pi) & \text{if } s = (((v)u, e), \pi) \end{cases}$$

Note that in the case where the current subterm is an abstraction and the stack is empty, the machine stops: it does not reduce under lambda's. That is why we slightly modify this machine in the following subsection.

1.2 A machine computing the principal head normal form

Now, the machine has to reduce under lambda's and, in Subsection 1.3, the machine will have to compute the arguments of the head variable. So, we extend the machine so that it performs the reduction of elements of $\mathcal{K}$, where $\mathcal{K} = \bigcup_{n \in \mathbb{N}} \mathcal{K}_n$ with

- $\mathcal{H}_0 = \mathcal{V}$ and $\mathcal{K}_0 = \mathbb{S}$;
- $\mathcal{H}_{n+1} = \mathcal{V} \cup \{(v)u \,/\, v \in \mathcal{H}_n \text{ and } u \in \Lambda \cup \mathcal{K}_n\}$ and
  $\mathcal{K}_{n+1} = \mathbb{S} \cup \mathcal{H}_n \cup \{\lambda y.k \,/\, y \in \mathcal{V} \text{ and } k \in \mathcal{K}_n\}$.

Set $\mathcal{H} = \bigcup_{n \in \mathbb{N}} \mathcal{H}_n$. We have $\mathcal{K} = \mathbb{S} \cup \mathcal{H} \cup \bigcup_{n \in \mathbb{N}} \{\lambda x.k \,/\, x \in \mathcal{V} \text{ and } k \in \mathcal{K}_n\}$.

Remark 3 *We have*

- $\mathcal{H} = \{(x)t_1 \ldots t_p \,/\, p \in \mathbb{N},\ x \in \mathcal{V},\ t_1, \ldots, t_p \in \Lambda \cup \mathcal{K}\}$
- *hence any element of* $\mathcal{K}$ *can be written as either*

$$\lambda x_1 \ldots \lambda x_m.s \text{ with } m \in \mathbb{N},\ x_1, \ldots, x_m \in \mathcal{V} \text{ and } s \in \mathbb{S}$$

or either

$$\lambda x_1 \ldots \lambda x_m.(x)t_1 \ldots t_p \text{ with } m, p \in \mathbb{N},\ x_1, \ldots, x_m \in \mathcal{V} \text{ and } t_1, \ldots, t_p \in \mathcal{K} \cup \Lambda.$$

For any $k \in \mathcal{K}$, we denote by $d(k)$ the least integer p such that $k \in \mathcal{K}_p$.

We extend the definition of $\overline{s}$ for $s \in \mathbb{S}$ to $\overline{k}$ for $k \in \mathcal{K}$. For that, we set $\overline{t} = t$ if $t \in \Lambda$. This definition is by induction on $d(k)$:

- if $d(k) = 0$, then $k \in \mathbb{S}$ and thus $\overline{k}$ is already defined;
- if $k \in \mathcal{H}$, then there are two cases:
  - if $k \in \mathcal{V}$, then $\overline{k}$ is already defined (it is k) ;
  - else, $k = (v)u$ and we set $\overline{k} = (\overline{v})\overline{u}$;
- if $k = \lambda x.k_0$, then $\overline{k} = \lambda x.\overline{k_0}$.

Definition 4 *We define a partial map from $\mathcal{K}$ to $\mathcal{K}$: for any $k, k' \in \mathcal{K}$, the notation $k \succ_h k'$ will mean that the map assigns k' to k. The value of the map at k is defined, by induction on $d(k)$, as follows:*

$$k \mapsto \begin{cases} s' & \textit{if } k \in \mathbb{S} \textit{ and } k \succ_{\mathbb{S}} s' \\ (x)\overline{c_1}\dots\overline{c_q} & \textit{if } k = ((x,e),(c_1,\dots,c_q)) \in \mathbb{S},\ x \in \mathcal{V} \textit{ and } x \notin \mathsf{dom}(e) \\ \lambda x.((u,e),\epsilon) & \textit{if } k = ((\lambda x.u,e),\epsilon) \in \mathbb{S} \\ \textit{not defined} & \textit{if } k \in \mathcal{H} \\ \lambda y.k'_0 & \textit{if } k = \lambda y.k_0 \textit{ and } k_0 \succ_h k'_0 \end{cases}$$

A difference with the original machine is that our machine reduces under lambda's.

We denote by ${\succ_h}^*$ the reflexive transitive closure of $\succ_h$. For any $k \in \mathcal{K}$, k is said to be a *Krivine normal form* if for any $k' \in \mathcal{K}$, we do not have $k \succ_h k'$.

Definition 5 *For any $k_0 \in \mathcal{K}$, we define $l_h(k_0) \in \mathbb{N} \cup \{\infty\}$ as follows: if there exist $k_1, \dots, k_n \in \mathcal{K}$ such that $k_i \succ_h k_{i+1}$ for $0 \leq i \leq n-1$ and k_n is a Krivine normal form, then we set $l_h(k_0) = n$, else we set $l_h(k_0) = \infty$.*

Proposition 6 *For any $s \in \mathbb{S}$, for any $k' \in \mathcal{K}$, if $s{\succ_h}^*k'$ and k' is a Krivine normal form, then k' is a λ-term in head normal form.*

PROOF. By induction on $l_h(s)$.

The base case is trivial, because we never have $l_h(s) = 0$.

The inductive step is divided into five cases.

- If $s = ((x,e),(c_1,\dots,c_q))$, $x \in \mathcal{V}$ and $x \notin \mathrm{dom}(e)$, then $s \succ_h (x)\overline{c_1}\dots\overline{c_q}$. But $(x)\overline{c_1}\dots\overline{c_q}$ is a Krivine normal form and $(x)\overline{c_1}\dots\overline{c_q}$ is a λ-term in head normal form.
- If $s = ((\lambda x.u,e),\pi)$ and π is the empty stack ϵ, then $k' = \lambda x.k''$ with $((u,e),\epsilon){\succ_h}^*k''$. Now, by induction hypothesis, k'' is a λ-term in head normal form, hence k' too is a λ-term in head normal form.
- If $s = ((x,e),(c_1,\dots,c_q))$, $x \in \mathcal{V}$ and $x \in \mathrm{dom}(e)$, then $s \succ_h (e(x),\pi)$. Now, $(e(x),\pi){\succ_h}^*k'$, hence, by induction hypothesis, k' is a λ-term in head normal form.
- If $s = ((\lambda x.u,e),c.\pi)$, then $s \succ_h ((u,\{(x,c)\} \cup e),\pi)$. Now, $((u,\{(x,c)\} \cup e),\pi) \succ_h k'$, hence, by induction hypothesis, k' is a λ-term in head normal form.
- If $s = (((v)u,e),\pi)$, then $s \succ_h ((v,e),(u,e).\pi)$. Now, $((v,e),(u,e).\pi){\succ_h}^*k'$, hence, by induction hypothesis, k' is a λ-term in head normal form.

	output	current subterm	environment	stack
		$(\lambda x.(x)x)\lambda y.y$	$\emptyset$	ϵ
1		$\lambda x.(x)x$	$\emptyset$	$(\lambda y.y, \emptyset)$
2		$(x)x$	$\{x \mapsto (\lambda y.y, \emptyset)\}$	ϵ
3		x	$\{x \mapsto (\lambda y.y, \emptyset)\}$	$(x, \{x \mapsto (\lambda y.y, \emptyset)\})$
4		$\lambda y.y$	$\emptyset$	$(x, \{x \mapsto (\lambda y.y, \emptyset)\})$
5		y	$\{y \mapsto (x, \{x \mapsto (\lambda y.y, \emptyset)\})\}$	ϵ
6		x	$\{x \mapsto (\lambda y.y, \emptyset)\}$	ϵ
7		$\lambda y.y$	$\emptyset$	ϵ
8	$\lambda y.$	y	$\emptyset$	ϵ
9	$\lambda y.y$			

Figure 1: Example of computation of the principal head normal form

□

Example 7 *Set* $s = (((\lambda x.(x)x)\lambda y.y, \emptyset), \epsilon)$*. We have* $l_h(s) = 9$*:*

$$
\begin{array}{rll}
s & \succ_h & ((\lambda x.(x)x, \emptyset),\ (\lambda y.y, \emptyset)) \\
& \succ_h & (((x)x, \{(x, (\lambda y.y, \emptyset))\}),\ \epsilon) \\
& \succ_h & ((x, \{(x, (\lambda y.y, \emptyset))\}),\ (x, \{(x, (\lambda y.y, \emptyset))\})) \\
& \succ_h & ((\lambda y.y, \emptyset),\ (x, \{(x, (\lambda y.y, \emptyset))\})) \\
& \succ_h & ((y, \{(y, (x, \{(x, (\lambda y.y, \emptyset))\}))\}),\ \epsilon) \\
& \succ_h & ((x, \{(x, (\lambda y.y, \emptyset))\}),\ \epsilon) \\
& \succ_h & ((\lambda y.y, \emptyset),\ \epsilon) \\
& \succ_h & \lambda y.((y, \emptyset),\ \epsilon) \\
& \succ_h & \lambda y.y
\end{array}
$$

We present the same computation in a more attractive way in Figure 1.

Lemma 8 *For any* $k, k' \in \mathcal{K}$*, if* $k \succ_h k'$*, then* $\overline{k} \to_h \overline{k'}$*, where* $\to_h$ *is the reflexive closure of the head reduction.*

Proof. There are two cases.

- If $k \in \mathbb{S}$, then there are five cases.
  - If $k = ((x, e), (c_1, \ldots, c_q))$, $x \in \mathcal{V}$ and $x \notin \mathrm{dom}(e)$, then $\overline{k} = (x)\overline{c_1} \ldots \overline{c_q}$ and $\overline{k'} = \overline{(x)c_1 \ldots c_q} = (x)\overline{c_1} \ldots \overline{c_q}$: we have $\overline{k} = \overline{k'}$.
  - If $k = ((\lambda x.u, e), \pi)$ and π is the empty stack ϵ, then $\overline{k} = \overline{(\lambda x.u)[e]} = \lambda x.u[e]$ (because k respects the variable convention) and $\overline{k'} = \overline{\lambda x.((u, e), \epsilon)} = \lambda x.u[e]$: we have $\overline{k} = \overline{k'}$.
  - If $k = ((x, e), (c_1, \ldots, c_q))$, $x \in \mathcal{V}$ and $x \in \mathrm{dom}(e)$, then $\overline{k} = \overline{e(x)}\overline{c_1} \ldots \overline{c_q}$ and $\overline{k'} = \overline{(e(x), (c_1, \ldots, c_q))} = \overline{e(x)}\overline{c_1} \ldots \overline{c_q}$: we have $\overline{k} = \overline{k'}$.

– If $k = ((\lambda x.u, e), (c, c_1, \ldots, c_q))$, then $\overline{k} = ((\lambda x.u)[e])\overline{c}\overline{c_1}\ldots\overline{c_q} = (\lambda x.u[e])\overline{c}\overline{c_1}\ldots\overline{c_q}$ (because k respects the variable convention) and $\overline{k'} = \overline{((u, \{(x,c)\} \cup e))}\overline{c_1}\ldots\overline{c_q}$. Now, $\overline{k}$ reduces in a single head reduction step to $\overline{k'}$.

– If $k = (((v)u, e), (c_1 \ldots c_q))$, then $\overline{k} = (((v)u)[e])\overline{c_1}\ldots\overline{c_q} = (v[e])u[e]\overline{c_1}\ldots\overline{c_q}$ and $\overline{k'} = \overline{((v, e), (u, e).(c_1, \ldots, c_q))} = (v[e])u[e]\overline{c_1}\ldots\overline{c_q}$: we have $\overline{k} = \overline{k'}$.

- Else, $k = \lambda y.k_0$; then $\overline{k} = \lambda y.\overline{k_0}$ and $\overline{k'} = \overline{\lambda y.k'_0} = \lambda y.\overline{k'_0}$ with $k_0 \succ_h k'_0$: we have $\overline{k_0} \rightarrow_h \overline{k'_0}$, hence $\overline{k} \rightarrow_h \overline{k'}$.

□

Theorem 9 *For any $k \in \mathcal{K}$, if $l_h(k)$ is finite, then $\overline{k}$ is head normalizable.*

PROOF. By induction on $l_h(k)$.

If $l_h(k) = 0$, then $k \in \mathcal{H}$, hence k can be written as $(x)t_1 \ldots t_p$ and thus $\overline{k}$ can be written $(x)\overline{t_1}\ldots\overline{t_p}$: it is a head normal form. Else, apply Lemma 8. □

For any head normalizable λ-term t, we denote by $h(t)$ the number of head reductions of t.

Theorem 10 *For any $s = ((t, e), \pi) \in \mathbb{S}$, if $\overline{s}$ is head normalizable, then $l_h(s)$ is finite.*

PROOF. By well-founded induction on $(h(\overline{s}), d(e), t)$.

If $h(\overline{s}) = 0$, $d(e) = 0$ and $t \in \mathcal{V}$, then we have $l_h(s) = 1$.

Else, there are five cases.

- In the case where $t \in \mathcal{V} \cap \mathrm{dom}(e)$, we have $s \succ_h (e(t), \pi)$. Set $s' = (e(t), \pi)$ and $e(t) = (t', e')$. We have $\overline{s} = \overline{s'}$ and $d(e') < d(e)$, thus we can apply the induction hypothesis: $l_h(s')$ is finite and thus $l_h(s) = l_h(s') + 1$ is finite.

- In the case where $t \in \mathcal{V}$ and $t \notin \mathrm{dom}(e)$, we have $l_h(s) = 1$.

- In the case where $t = (v)u$, we have $s \succ_h ((v, e), (u, e).\pi)$. Set $s' = ((v, e), (u, e).\pi)$. We have $\overline{s'} = \overline{s}$ and thus we can apply the induction hypothesis: $l_h(s')$ is finite and thus $l_h(s) = l_h(s') + 1$ is finite.

- In the case where $t = \lambda x.u$ and $\pi = \epsilon$, we have $s \succ_h \lambda x.((u, e), \epsilon)$. Set $s' = ((u, e), \epsilon)$. Since s respects the variable convention, we have $\overline{s} = \lambda x.u[e] = \lambda x.\overline{s'}$. We have $h(\overline{s'}) = h(\overline{s})$, hence we can apply the induction hypothesis: $l_h(s')$ is finite and thus $l_h(s) = l_h(s') + 1$ is finite.

- In the case where $t = \lambda x.u$ and $\pi = c.\pi'$, we have $s \succ_h ((u, \{(x, c)\} \cup e), \pi)$. Set $s' = ((u, \{(x, c)\} \cup e), \pi)$. We have $h(\overline{s'}) < h(\overline{s})$, hence we can apply the induction hypothesis: $l_h(s')$ is finite and thus $l_h(s) = l_h(s') + 1$ is finite.

□

We recall that if a λ-term t has a head-normal form, then the last term of the terminating head reduction of t is called *the principal head normal form of t* (see [Barendregt 1984]). Proposition 6, Lemma 8 and Theorem 10 show that for any head normalizable λ-term t with t' its principal head normal form, we have $((t, \emptyset), \epsilon) {\succ_h}^* t'$ and t' is a Krivine head normal form.

1.3 A machine computing the β-normal form

We now slightly modify the machine so as to compute the β-normal form of any normalizable λ-term.

Definition 11 *We define a partial map from $\mathcal{K}$ to $\mathcal{K}$: for any $k, k' \in \mathcal{K}$, the notation $k \succ_\beta k'$ will mean that the map assigns k' to k. The value of the map at k is defined, by induction on $d(k)$, as follows:*

$$k \mapsto \begin{cases} s' & \textit{if } k \in \mathbb{S} \textit{ and } k \succ_{\mathbb{S}} s' \\ (x)(c_1,\epsilon)\ldots(c_q,\epsilon) & \textit{if } k = ((x,e),(c_1,\ldots,c_q)) \in \mathbb{S},\ x \in \mathcal{V} \textit{ and } x \notin \mathsf{dom}(e) \\ \lambda x.((u,e),\epsilon) & \textit{if } k = ((\lambda x.u,e),\epsilon) \in \mathbb{S} \\ \textit{not defined} & \textit{if } k \in \mathcal{V} \\ (v')u & \textit{if } k = (v)u \textit{ and } v \succ_\beta v' \\ (x)u' & \textit{if } k = (x)u \textit{ with } x \in \mathcal{V} \textit{ and } u \succ_\beta u' \\ \lambda y.k_0' & \textit{if } k = \lambda y.k_0 \textit{ and } k_0 \succ_\beta k_0' \end{cases}$$

Let us compare Definition 11 with Definition 4. The difference is in the case where the current subterm of a state is a variable and where this variable has no value in the environment: the first machine stops, the second machine continues to compute every argument of the variable.

The function l_β is defined as l_h (see Definition 5), but for this new machine.

For any normalizable λ-term t, we denote by $n(t)$ the number of left reductions of t.

Theorem 12 *For any $s = ((t,e),\pi) \in \mathbb{S}$, if $\overline{s}$ is normalizable, then $l_\beta(s)$ is finite.*

Proof. By well-founded induction on $(n(\overline{s}), \overline{s}, d(e), t)$.

If $n(\overline{s}) = 0$, $\overline{s} \in \mathcal{V}$, $d(e) = 0$ and $t \in \mathcal{V}$, then we have $l_\beta(s) = 1$.

Else, there are five cases.

- In the case where $t \in \mathcal{V} \cap \mathrm{dom}(e)$, we have $s \succ_\beta (e(t),\pi)$. Set $s' = (e(t),\pi)$ and $e(t) = (t',e')$. We have $\overline{s} = \overline{s'}$ and $d(e') < d(e)$, hence we can apply the induction hypothesis: $l_\beta(s')$ is finite and thus $l_\beta(s) = l_\beta(s') + 1$ is finite.

- In the case where $t \in \mathcal{V}$ and $t \notin \mathrm{dom}(e)$, set $\pi = (c_1,\ldots,c_q)$. For any $k \in \{1,\ldots,q\}$, we have $n(\overline{c_k}) \leq n(\overline{s})$ and $\overline{c_k} < \overline{s}$, hence we can apply the induction hypothesis on c_k: for any $k \in \{1,\ldots,q\}$, $l_\beta(c_k)$ is finite, hence $l_\beta(s) = \sum_{k=1}^{q} l_\beta(c_k) + 1$ is finite too.

- In the case where $t = (v)u$, we have $s \succ_\beta ((v,e),(u,e).\pi)$. Set $s' = ((v,e),(u,e).\pi)$. We have $\overline{s'} = \overline{s}$, hence we can apply the induction hypothesis: $l_\beta(s')$ is finite and thus $l_\beta(s) = l_\beta(s') + 1$ is finite.

- In the case where $t = \lambda x.u$ and $\pi = \epsilon$, we have $s \succ_\beta \lambda x.((u,e),\epsilon)$. Set $s' = ((u,e),\epsilon)$. Since s respects the variable convention, we have $\overline{s} = \lambda x.u[e] = \lambda x.\overline{s'}$. We have $n(\overline{s'}) = n(\overline{s})$, hence we can apply the induction hypothesis: $l_\beta(s')$ is finite and thus $l_\beta(s) = l_\beta(s') + 1$ is finite.

- In the case where $t = \lambda x.u$ and $\pi = c.\pi'$, we have $s \succ_\beta ((u,\{(x,c)\} \cup e),\pi)$. Set $s' = ((u,\{(x,c)\} \cup e),\pi)$. We have $n(\overline{s'}) < n(\overline{s})$, hence we can apply the induction hypothesis: $l_\beta(s')$ is finite and thus $l_\beta(s) = l_\beta(s') + 1$ is finite.

$\square$

2 A non-uniform semantics

We define here the semantics allowing to measure execution time. We have in mind the following philosophy: the semantics for the untyped λ-calculus come from the semantics for the typed λ-calculus and any semantics for linear logic induces a semantics for the typed λ-calculus. So, we start from a semantics $\mathfrak{M}$ for linear logic (Subsection 2.1), then we present the induced semantics $\Lambda(\mathfrak{M})$ for the typed λ-calculus (Subsection 2.2) and lastly the semantics of the untyped λ-calculus that we consider (Subsection 2.3). This semantics is *non-uniform*: in Subsection 2.4, we give an example for illustrating this point.

The first works tackling the problem of giving a general categorical definition of a denotational model of linear logic are those of Lafont [Lafont 1988] and of Seely [Seely 1989]. As for the works of Benton, Bierman, Hyland and de Paiva, [Benton *et al.* 1994], [Bierman 1993] and [Bierman 1995], they led to the following axiomatic: a categorical model of the multiplicative exponential fragment of intuitionistic linear logic (IMELL) is a quadruple $(\mathcal{C}, \mathcal{L}, c, w)$ such that

- $\mathcal{C} = (\mathbb{C}, \otimes, I, \alpha, \lambda, \rho, \gamma)$ is a closed symmetric monoidal category;
- $\mathcal{L} = ((T, \mathsf{m}, \mathsf{n}), \delta, d)$ is a symmetric monoidal comonad on $\mathcal{C}$;
- c is a monoidal natural transformation from $(T, \mathsf{m}, \mathsf{n})$ to $\otimes \circ \Delta_{\mathcal{C}} \circ (T, \mathsf{m}, \mathsf{n})$ and w is a monoidal natural transformation from $(T, \mathsf{m}, \mathsf{n})$ to $*_{\mathcal{C}}$ such that
  - for any object A of $\mathbb{C}$, $((T(A), \delta_A), c_A, w_A)$ is a cocommutative comonoid in $(\mathbb{C}^{\mathbb{T}}, \otimes^{\mathbb{T}}, (I, \mathsf{n}), \alpha, \lambda, \rho)$
  - and for any $f \in \mathbb{C}^{\mathbb{T}}[(T(A), \delta_A), (T(B), \delta_B)]$, f is a comonoid morphism,

  where $\mathbb{T}$ is the comonad (T, δ, d) on $\mathbb{C}$, $\mathbb{C}^{\mathbb{T}}$ is the category of $\mathbb{T}$-coalgebras, $\Delta_{\mathcal{C}}$ is the diagonal monoidal functor from $\mathcal{C}$ to $\mathcal{C} \times \mathcal{C}$ and $*_{\mathcal{C}}$ is the monoidal functor that sends any arrow to id_I.

Given a categorical model $\mathfrak{M} = (\mathcal{C}, \mathcal{L}, c, w)$ of IMELL with $\mathcal{C} = (\mathbb{C}, \otimes, I, \alpha, \lambda, \rho, \gamma)$ and $\mathcal{L} = ((T, \mathsf{m}, \mathsf{n}), \delta, d)$, we can define a cartesian closed category $\Lambda(\mathfrak{M})$ such that

- the objects are finite sequences of objects of $\mathbb{C}$
- and the arrows $\langle A_1, \ldots, A_m \rangle \to \langle B_1, \ldots, B_p \rangle$ are sequences $\langle f_1, \ldots, f_p \rangle$ where every f_k is an arrow $\bigotimes_{j=1}^{m} T(A_j) \to B_k$ in $\mathbb{C}$.

Hence we can interpret simply typed λ-calculus in the category $\Lambda(\mathfrak{M})$. This category is (weakly) equivalent[1] to a full subcategory of (T, δ, d)-coalgebras exhibited by Hyland. If the category $\mathbb{C}$ is cartesian, then the categories $\Lambda(\mathfrak{M})$ and the Kleisli category of the comonad (T, δ, d) are (strongly) equivalent[2]. See [de Carvalho 2007] for a full exposition.

2.1 A relational model for linear logic

The category of sets and relations is denoted by **Rel** and $\circ$ denotes its composition. The functor T from **Rel** to **Rel** is defined by setting

[1] A category $\mathbb{C}$ is said to be *weakly equivalent* to a category $\mathbb{D}$ if there exists a functor $F : \mathbb{C} \to \mathbb{D}$ full and faithful such that every object D of $\mathbb{D}$ is isomorphic to $F(C)$ for some object C of $\mathbb{C}$.

[2] A category $\mathbb{C}$ is said to be *strongly equivalent* to a category $\mathbb{D}$ if there are functors $F : \mathbb{C} \to \mathbb{D}$ and $G : \mathbb{D} \to \mathbb{C}$ and natural isomorphisms $G \circ F \cong id_{\mathbb{C}}$ and $F \circ G \cong id_{\mathbb{D}}$.

- for any object A of **Rel**, $T(A) = \mathcal{M}_f(A)$, the set of finite multisets a whose support, denoted by $\mathsf{Supp}(a)$, is a subset of A;
- and, for any $f \in \mathbf{Rel}(A, B)$, $T(f) \in \mathbf{Rel}(T(A), T(B))$ defined by

$$T(f) = \{([\alpha_1, \ldots, \alpha_n], [\beta_1, \ldots, \beta_n]) \mathrel{/} n \in \mathbb{N} \text{ and } (\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n) \in f\} \ .$$

The natural transformation d from T to the identity functor of **Rel** is defined by setting $d_A = \{([\alpha], \alpha) \mathrel{/} \alpha \in A\}$ and the natural transformation δ from T to $T \circ T$ by setting $\delta_A = \{(a_1 + \ldots + a_n, [a_1, \ldots, a_n]) \mathrel{/} n \in \mathbb{N} \text{ and } a_1, \ldots, a_n \in T(A)\}$. It is easy to show that (T, δ, d) is a comonad on **Rel**. It is well-known that this comonad can be provided with a structure $\mathfrak{M}$ that provides a model of (I)MELL.

This model gives rise to a cartesian closed category $\Lambda(\mathfrak{M})$.

2.2 Interpreting simply typed λ-terms

We describe the category $\Lambda(\mathfrak{M})$ induced by the model $\mathfrak{M}$ of linear logic presented in the preceding subsection:

- objects are finite sequences of sets;
- arrows $\langle A_1, \ldots, A_m \rangle \to \langle B_1, \ldots, B_n \rangle$ are sequences $\langle f_1, \ldots, f_n \rangle$ where every f_i is a subset of $(\prod_{j=1}^m \mathcal{M}_f(A_j)) \times B_i$ with the convention $(\prod_{j=1}^m \mathcal{M}_f(A_j)) \times B_i = B_i$ if $m = 0$;
- if $\langle f_1, \ldots, f_p \rangle$ is an arrow $\langle A_1, \ldots, A_m \rangle \to \langle B_1, \ldots, B_p \rangle$ and $\langle g_1, \ldots, g_q \rangle$ is an arrow $\langle B_1, \ldots, B_p \rangle \to \langle C_1, \ldots, C_q \rangle$, then $\langle g_1, \ldots, g_q \rangle \circ_{\Lambda(\mathfrak{M})} \langle f_1, \ldots, f_p \rangle$ is $\langle h_1, \ldots, h_q \rangle$ with

$$h_l = \left\{ \begin{array}{l} ((\sum_{k=1}^p \sum_{i=1}^{n_k} a_1^{i,k}, \ldots, \sum_{k=1}^p \sum_{i=1}^{n_k} a_m^{i,k}), \gamma) \mathrel{/} n_1, \ldots, n_p \in \mathbb{N} \text{ and} \\ \quad \text{for } 1 \leq j \leq m, \text{ for } 1 \leq k \leq p, \text{ for } 1 \leq i \leq n_k, a_j^{i,k} \in \mathcal{M}_f(A_j) \text{ s.t.} \\ \quad\quad \exists \beta_1^1, \ldots, \beta_1^{n_1} \in B_1, \ldots, \exists \beta_p^1, \ldots, \beta_p^{n_p} \in B_p \text{ s.t.} \\ \quad\quad\quad (([\beta_1^1, \ldots, \beta_1^{n_1}], \ldots, [\beta_p^1, \ldots, \beta_p^{n_p}]), \gamma) \in g_l \text{ and} \\ \quad\quad\quad \text{for } 1 \leq k \leq p, \text{ for } 1 \leq i \leq n_k, ((a_1^{i,k}, \ldots, a_m^{i,k}), \beta_k^i) \in f_k \end{array} \right\}$$

  for $1 \leq l \leq q$, with the conventions

$$((a_1, \ldots, a_m), \gamma) = \gamma \text{ and } (\prod_{j=1}^m \mathcal{M}_f(A_j)) \times C_l = C_l \text{ if } m = 0 \ .$$

- the identity of $\langle A_1, \ldots, A_m \rangle$ is $\langle d^1, \ldots, d^m \rangle$ with

$$d^j = \{((\underbrace{[], \ldots, []}_{j-1 \text{ times}}, [\alpha], \underbrace{[], \ldots, []}_{m-j \text{ times}}), \alpha) \mathrel{/} \alpha \in A_j\} \ .$$

The category $\Lambda(\mathfrak{M})$ has the following cartesian closed structure

$$(\Lambda(\mathfrak{M}), 1, !, \&, \pi^1, \pi^2, \langle \cdot, \cdot \rangle_{\mathfrak{M}}, \Rightarrow, \Lambda, \mathrm{ev}) :$$

- the terminal object 1 is the empty sequence $\langle \rangle$;
- if $B^1 = \langle B_1, \ldots, B_p \rangle$ and $B^2 = \langle B_{p+1}, \ldots, B_{p+q} \rangle$ are two sequences of sets, then $B^1 \& B^2$ is the sequence $\langle B_1, \ldots, B_{p+q} \rangle$;

- if $B^1 = \langle B_1, \ldots, B_p \rangle$ and $B^2 = \langle B_{p+1}, \ldots, B_{p+q} \rangle$ are two sequences of sets, then
$$\pi^1_{B^1,B^2} = \langle d^1, \ldots, d^p \rangle : B^1 \& B^2 \to B^1 \text{ in } \Lambda(\mathfrak{M})$$
and
$$\pi^2_{B^1,B^2} = \langle d^{p+1}, \ldots, d^{p+q} \rangle : B^1 \& B^2 \to B^2 \text{ in } \Lambda(\mathfrak{M})$$
with
$$d^k = \{((\underbrace{[], \ldots, []}_{k-1 \text{ times}}, [\beta], \underbrace{[], \ldots, []}_{p+q-k \text{ times}}), \beta) \;/\; \beta \in B_k\} \;;$$

- if $f^1 = \langle f_1, \ldots, f_p \rangle : C \to A^1$ and $f^2 = \langle f_{p+1}, \ldots, f_{p+q} \rangle : C \to A^2$ in $\Lambda(\mathfrak{M})$, then $\langle f^1, f^2 \rangle_{\mathfrak{M}} = \langle f_1, \ldots, f_{p+q} \rangle : C \to A^1 \& A^2$;

and

- $\langle A_1, \ldots, A_m \rangle \Rightarrow \langle C_1, \ldots, C_q \rangle$ is defined by induction on m:
  - $\langle\rangle \Rightarrow \langle C_1, \ldots, C_q \rangle = \langle C_1, \ldots, C_q \rangle$
  -
$$\begin{aligned} & \langle A_1, \ldots, A_{m+1} \rangle \Rightarrow \langle C_1, \ldots, C_q \rangle \\ = \; & \langle \langle A_1, \ldots, A_m \rangle \Rightarrow (\mathcal{M}_f(A_{m+1}) \times C_1), \ldots, \\ & \langle A_1, \ldots, A_m \rangle \Rightarrow (\mathcal{M}_f(A_{m+1}) \times C_q) \rangle \;; \end{aligned}$$

- if $h = \langle h_1, \ldots, h_q \rangle : \langle A_1, \ldots, A_m \rangle \& \langle B_1, \ldots, B_p \rangle \to \langle C_1, \ldots, C_q \rangle$, then
$$\Lambda^{\langle B_1, \ldots, B_p \rangle}_{\langle A_1, \ldots, A_m \rangle, \langle C_1, \ldots, C_q \rangle}(h) : \langle A_1, \ldots, A_m \rangle \to \langle B_1 \ldots, B_p \rangle \Rightarrow \langle C_1, \ldots, C_q \rangle$$
is defined by induction on p:
  - if $p = 0$, then $\Lambda^{\langle B_1, \ldots, B_p \rangle}_{\langle A_1, \ldots, A_m \rangle, \langle C_1, \ldots, C_q \rangle}(h) = h$;
  - if $p = 1$, then there are two cases:
    * in the case $m = 0$, $\Lambda^{\langle B_1, \ldots, B_p \rangle}_{\langle A_1, \ldots, A_m \rangle, \langle C_1, \ldots, C_q \rangle}(h) = h$;
    * in the case $m \neq 0$,
$$\Lambda^{\langle B_1, \ldots, B_p \rangle}_{\langle A_1, \ldots, A_m \rangle, \langle C_1, \ldots, C_q \rangle}(h) = \langle \xi^{\mathcal{M}_f(B_1)}_{\prod_{j=1}^m \mathcal{M}_f(A_j), C_1}(h_1), \ldots, \xi^{\mathcal{M}_f(B_1)}_{\prod_{j=1}^m \mathcal{M}_f(A_j), C_q}(h_q) \rangle \;,$$
where
$$\xi^{\mathcal{M}_f(B_1)}_{\prod_{j=1}^m \mathcal{M}_f(A_j), C_l}(h_l) = \{(a, (b, \gamma)) \;;\; ((a, b), \gamma) \in h_l\} \;;$$
  - if $p \geq 1$, then
$$\begin{aligned} & \Lambda^{\langle B_1, \ldots, B_{p+1} \rangle}_{A, \langle C_1, \ldots, C_q \rangle}(h) \\ = \; & \Lambda^{\langle B_1, \ldots, B_p \rangle}_{A, \langle \mathcal{M}_f(B_{p+1}) \times C_1, \ldots, \mathcal{M}_f(B_{p+1}) \times C_q \rangle}(\Lambda^{\langle B_{p+1} \rangle}_{\langle A_1, \ldots, A_m, B_1, \ldots, B_p \rangle, \langle C_1, \ldots, C_q \rangle}(h)) \;, \end{aligned}$$
where $A = \langle A_1, \ldots, A_m \rangle$;

- $\mathrm{ev}_{C,B} : (B \Rightarrow C)\&B \to C$ is defined by setting

$$\mathrm{ev}_{\langle C_1,\ldots,C_q\rangle,\langle B_1,\ldots,B_p\rangle} = \langle \mathrm{ev}^1_{\langle C_1,\ldots,C_q\rangle,\langle B_1,\ldots,B_p\rangle}, \ldots, \mathrm{ev}^q_{\langle C_1,\ldots,C_q\rangle,\langle B_1,\ldots,B_p\rangle}\rangle$$

where, for $1 \leq k \leq q$,

$$\mathrm{ev}^k_{\langle C_1,\ldots,C_q\rangle,\langle B_1,\ldots,B_p\rangle} = \left\{ \begin{array}{l} ((\underbrace{[],\ldots,[]}_{k-1 \text{ times}}, [((b_1,\ldots,b_p),\gamma)], \underbrace{[],\ldots,[]}_{q-k \text{ times}}, b_1,\ldots,b_p),\gamma) \; / \\ b_1 \in \mathcal{M}_f(B_1),\ldots,b_p \in \mathcal{M}_f(B_p), \gamma \in C_k \end{array} \right\} .$$

2.3 Interpreting type free λ-terms

With the cartesian closed structure on $\Lambda(\mathfrak{M})$, we have a semantics of the simply typed λ-calculus. Now, in order to have a semantics of the pure λ-calculus, it is therefore enough to have a *reflexive* object U of $\Lambda(\mathfrak{M})$, that is to say such that

$$(U \Rightarrow U) \lhd U \ ,$$

that means that there exist $s \in \Lambda(\mathfrak{M})[U \Rightarrow U, U]$ and $r \in \Lambda(\mathfrak{M})[U, U \Rightarrow U]$ such that $r \circ_{\Lambda(\mathfrak{M})} s$ is the identity on $U \Rightarrow U$. We will use the following lemma. We recall that $\langle f\rangle$ *is a retraction of* $\langle g\rangle$ *in* $\Lambda(\mathfrak{M})$ means that $\langle f\rangle \circ_{\Lambda(\mathfrak{M})} \langle g\rangle = id_{\langle A\rangle}$ (see, for instance, [Mac Lane 1998]). It is also said that $(\langle g\rangle, \langle f\rangle)$ *is a retraction pair.*

Lemma 13 *Let $h : A \to B$ be an injection between sets. Set*

$$g = \{([\alpha], h(\alpha)) : \alpha \in A\} : \mathcal{M}_f(A) \to B \text{ in } \mathbf{Rel}$$

and

$$f = \{([h(\alpha)], \alpha)/\alpha \in A\} : \mathcal{M}_f(B) \to A \text{ in } \mathbf{Rel} \ .$$

Then $\langle g\rangle \in \Lambda(\mathfrak{M})(\langle A\rangle, \langle B\rangle)$ and $\langle f\rangle$ is a retraction of $\langle g\rangle$ in $\Lambda(\mathfrak{M})$.

Proof. An easy computation shows that we have

$$\begin{array}{rcl} \langle f\rangle \circ_{\Lambda(\mathfrak{M})} \langle g\rangle & = & \langle f \circ T(g) \circ \delta_A\rangle \\ & = & \langle d_A\rangle \ . \end{array}$$

□

If D is a set, then $\langle D\rangle \Rightarrow \langle D\rangle = \langle \mathcal{M}_f(D) \times D\rangle$. From now on, we assume that D is a non-empty set and that h is an injection from $\mathcal{M}_f(D) \times D$ to D. Set

$$g = \{([\alpha], h(\alpha)) \;/\; \alpha \in \mathcal{M}_f(D) \times D\} : \mathcal{M}_f(\mathcal{M}_f(D) \times D) \to D \text{ in } \mathbf{Rel}$$

and

$$f = \{([h(\alpha)], \alpha) \;/\; \alpha \in \mathcal{M}_f(D) \times D\} : \mathcal{M}_f(D) \to \mathcal{M}_f(D) \times D \text{ in } \mathbf{Rel} \ .$$

We have

$$(\langle D\rangle \Rightarrow \langle D\rangle) \lhd \langle D\rangle$$

and, more precisely, $\langle g\rangle \in \Lambda(\mathfrak{M})(\langle D\rangle \Rightarrow \langle D\rangle, \langle D\rangle)$ and f is a retraction of g.

We can therefore define the interpretation of any λ-term.

Definition 14 *For any λ-term t possibly containing constants from $\mathcal{P}(D)$, for any $x_1, \ldots, x_m \in \mathcal{V}$ distinct such that $FV(t) \subseteq \{x_1, \ldots, x_m\}$, we define, by induction on t, $[\![t]\!]_{x_1,\ldots,x_m} \subseteq (\prod_{j=1}^{m} \mathcal{M}_f(D)) \times D$:*

- $[\![x_j]\!]_{x_1,\ldots,x_m} = \{((\underbrace{[],\ldots,[]}_{j-1 \ times}, [\alpha], \underbrace{[],\ldots,[]}_{m-j \ times}), \alpha) \ / \ \alpha \in D\}$;
- *for any $c \in \mathcal{P}(D)$, $[\![c]\!]_{x_1,\ldots,x_m} = (\prod_{j=1}^{m} \mathcal{M}_f(D)) \times c$;*
- $[\![\lambda x.u]\!]_{x_1,\ldots,x_m} = \{((a_1, \ldots, a_m), h(a, \alpha)) \ / \ ((a_1, \ldots, a_m, a), \alpha) \in [\![u]\!]_{x_1,\ldots,x_m,x}\}$;
- $$[\![(v)u]\!]_{x_1,\ldots,x_m} = \left\{ \begin{array}{c} ((\sum_{i=0}^{n} a_1^i, \ldots, \sum_{i=0}^{n} a_m^i), \alpha) \ / \ \exists(\alpha_1, \ldots, \alpha_n) \ s.t. \\ ((a_1^0, \ldots, a_m^0), h([\alpha_1, \ldots, \alpha_n], \alpha)) \in [\![v]\!]_{x_1,\ldots,x_m} \\ and, \ for \ 1 \leq i \leq n, ((a_1^i, \ldots, a_m^i), \alpha_i) \in [\![u]\!]_{x_1,\ldots,x_m}) \end{array} \right\} ;$$

with the conventions $(\prod_{j=1}^{m} \mathcal{M}_f(D)) \times D = D$ and $((a_1, \ldots, a_m), \alpha) = \alpha$ if $m = 0$.

Now, we can define the interpretation of any λ-term in any environment.

Definition 15 *For any $\rho \in \mathcal{P}(D)^{\mathcal{V}}$ and for any λ-term t possibly containing constants from $\mathcal{P}(D)$ such that $FV(t) = \{x_1, \ldots, x_m\}$, we set*

$$[\![t]\!]_\rho = \{\alpha \in D \ / \ ((a_1, \ldots, a_m), \alpha) \in [\![t]\!]_{x_1,\ldots,x_m} \ and, \ for \ 1 \leq j \leq m, \ a_i \in \mathcal{M}_f(\rho(x_j))\} \ .$$

For any $d_1, d_2 \in \mathcal{P}(D)$, we set

$$d_1 * d_2 = \{\alpha \in D \ / \ \exists a \ (h(a, \alpha)) \in d_1 \text{ and } \mathsf{Supp}(a) \subseteq d_2\} \ .$$

The triple $(\mathcal{P}(D), *, [\![-]\!]_-)$ is a λ-algebra (Theorem 5.5.6 of [Barendregt 1984]). But the following proposition, a corollary of Proposition 17, states that it *is not* a λ-model. We recall that a λ-model is a λ-algebra $(\mathcal{D}, *, [\![-]\!]_-)$ such that the following property, expressing the ξ-rule, holds:
for any $\rho \in \mathcal{D}^{\mathcal{V}}$, for any $x \in \mathcal{V}$ and for any λ-terms t_1 and t_2, we have

$$(\forall d \in \mathcal{D} \ [\![t_1]\!]_{\rho[x:=d]} = [\![t_2]\!]_{\rho[x:=d]} \Rightarrow [\![\lambda x.t_1]\!]_\rho = [\![\lambda x.t_2]\!]_\rho) \ .$$

Proposition 16 *The λ-algebra $(\mathcal{P}(D), *, [\![-]\!]_-)$ is not a λ-model.*
In other words, there exist $\rho \in \mathcal{P}(D)^{\mathcal{V}}$, $x \in \mathcal{V}$ and two λ-terms t_1 and t_2 such that

$$(\forall d \in \mathcal{P}(D) \ [\![t_1]\!]_{\rho[x:=d]} = [\![t_2]\!]_{\rho[x:=d]} \ and \ [\![\lambda x.t_1]\!]_\rho \neq [\![\lambda x.t_2]\!]_\rho) \ .$$

In particular, $[\![t]\!]_\rho$ *can not be* defined by induction on t (an interpretation by polynomials is nevertheless possible in such a way that the ξ-rule holds - see [Selinger 2002]).

Before stating Proposition 17, we recall that any object A of any category $\mathbb{K}$ with a terminal object is said *to have enough points* if for any terminal object 1 of $\mathbb{K}$ and for any $y, z \in \mathbb{K}(A, A)$, we have $(\forall x \in \mathbb{K}(1, A) \ y \circ_{\mathbb{K}} x = z \circ_{\mathbb{K}} x \Rightarrow y = z)$.

Remark: it does not follow necessarily that the same holds for any $y, z \in \mathbb{K}(A, B)$.

Proposition 17 *Let A be a non-empty set. Then $\langle A \rangle$ does not have enough points in $\Lambda(\mathfrak{M})$.*

PROOF. Let $\alpha \in A$. Set

$$y = \{([\alpha], \alpha)\} : \mathcal{M}_f(A) \to A \text{ in } \mathbf{Rel}$$

and

$$z = \{([\alpha, \alpha], \alpha)\} : \mathcal{M}_f(A) \to A \text{ in } \mathbf{Rel} \ .$$

We have $\langle y \rangle$, $\langle z \rangle : \langle A \rangle \to \langle A \rangle$ in $\Lambda(\mathfrak{M})$.

We recall that the terminal object in $\Lambda(\mathfrak{M})$ is the empty sequence $\langle\rangle$. Now, for any $x : \langle\rangle \to \langle A \rangle$ in $\Lambda(\mathfrak{M})$, we have $\langle y \rangle \circ_{\Lambda(\mathfrak{M})} x = \langle z \rangle \circ_{\Lambda(\mathfrak{M})} x$. $\square$

This proposition explains *why* Proposition 16 holds. A more direct proof of Proposition 16 consists by considering the two λ-terms $t_1 = (y)x$ and $t_2 = (z)x$ with $\rho(y) = \{([\alpha], \alpha)\}$ and $\rho(z) = \{([\alpha, \alpha], \alpha)\}$.

2.4 Non-uniformity

Example 18 illustrates the non-uniformity of the semantics. It is based on the following idea.

Consider the program

```
λx.if x  then 1
         else if x  then 1
                    else 0
```

applied to a boolean. The second then is never read. A *uniform* semantics would ignore it. It is not the case when the semantics is *non-uniform*.

Example 18 *Set* $\mathbf{0} = \lambda x.\lambda y.y$ *and* $\mathbf{1} = \lambda x.\lambda y.x$. *Assume that* h *is the inclusion from* $\mathcal{M}_f(D) \times D$ *to* D.

Let $\gamma \in D$*; set* $\delta = ([], ([\gamma], \gamma))$ *and* $\beta = ([\gamma], ([], \gamma))$. *We have*

- $([([], ([\delta], \delta))], ([\delta], \delta)) \in [\![(x)\mathbf{1}]\!]_x$*;*
- *and* $([([], ([\delta], \delta))], \delta) \in [\![(x)\mathbf{10}]\!]_x$.

Hence we have $\alpha_1 = ([([], ([\delta], \delta)), ([], ([\delta], \delta))], \delta) \in [\![\lambda x.(x)\mathbf{1}(x)\mathbf{10}]\!]$.

We have

- $([([], ([\beta], \beta))], ([\beta], \beta)) \in [\![(x)\mathbf{1}]\!]_x$*;*
- *and* $([([\beta], ([], \beta))], \beta) \in [\![(x)\mathbf{10}]\!]_x$.

Hence we have $\alpha_2 = ([([], ([\beta], \beta)), ([\beta], ([], \beta))], \beta) \in [\![\lambda x.(x)\mathbf{1}(x)\mathbf{10}]\!]$.

In an uniform semantics (as in [Girard 1986]), the point α_1 *would appear in the semantics of this* λ*-term, but not the point* α_2*, because* $[([], ([\beta], \beta)), ([\beta], ([], \beta))]$ *corresponds to a chimerical argument: the argument is read twice and provides two contradictory values.*

3 Non-idempotent intersection types

From now on, $D = \bigcup_{n \in \mathbb{N}} D_n$, where D_n is defined by induction on n: D_0 is a non-empty set A that does not contain any pairs and $D_{n+1} = A \cup (\mathcal{M}_f(D_n) \times D_n)$. We have $D = A \dot{\cup} (\mathcal{M}_f(D) \times D)$, where $\dot{\cup}$ is the disjoint union; the injection h from $\mathcal{M}_f(D) \times D$ to D will be the inclusion. Hence any element of D can be written $a_1 \ldots a_m \alpha$, where $a_1, \ldots, a_m \in \mathcal{M}_f(D)$, $\alpha \in D$ and $a_1 \ldots a_m \alpha$ is defined by induction on m:

- $a_1 \dots a_0 \alpha = \alpha$;
- $a_1 \dots a_{m+1} \alpha = (a_1 \dots a_m, (a_{m+1}, \alpha))$.

In the preceding section, we defined the semantics we consider: Definition 14 defines $[\![t]\!]_{x_1,\dots,x_m}$ for any λ-term t and for any $x_1, \dots, x_m \in \mathcal{V}$ distinct such that $FV(t) \subseteq \{x_1, \dots, x_m\}$; Definition 15 defines $[\![t]\!]_\rho$ for any λ-term t and for any $\rho \in \mathcal{P}(D)^{\mathcal{V}}$. Now, we want to put this semantics in a logical framework: the elements of D are viewed as propositional formulas. More precisely, a comma separating a multiset of types and a type is understood as an arrow and a non-empty multiset is understood as the uniform conjunction of its elements (their intersection). Note that this means we are considering a commutative but not necessarily idempotent intersection.

3.1 System R

A *context* Γ is a function from $\mathcal{V}$ to $\mathcal{M}_f(D)$ such that $\{x \in \mathcal{V} \ / \ \Gamma(x) \neq []\}$ is finite. If $x_1, \dots, x_m \in \mathcal{V}$ are distinct and $a_1, \dots, a_m \in \mathcal{M}_f(D)$, then $x_1 : a_1, \dots, x_m : a_m$ denotes the context defined by $x \mapsto \begin{cases} a_j & \text{if } x = x_j \\ [] & \text{else} \end{cases}$. We denote by Φ the set of contexts. For $\Gamma_1, \Gamma_2 \in \Phi$, $\Gamma_1 + \Gamma_2$ is the context defined by $(\Gamma_1 + \Gamma_2)(x) = \Gamma_1(x) + \Gamma_2(x)$, where the second $+$ denotes the sum of multisets given by term-by-term addition of multiplicities.

Typing rules concern judgements of the form $\Gamma \vdash_R t : \alpha$, where Γ is a context, t is a λ-term and $\alpha \in D$.

Definition 19 *The typing rules of System R are the following:*

$$\frac{}{x : [\alpha] \vdash_R x : \alpha}$$

$$\frac{\Gamma, x : a \vdash_R v : \alpha}{\Gamma \vdash_R \lambda x.v : (a, \alpha)}$$

$$\frac{\Gamma_0 \vdash_R v : ([\alpha_1, \dots, \alpha_n], \alpha) \qquad \Gamma_1 \vdash_R u : \alpha_1, \dots, \Gamma_n \vdash_R u : \alpha_n}{\Gamma_0 + \Gamma_1 + \dots + \Gamma_n \vdash_R (v)u : \alpha} \; n \in \mathbb{N}$$

The typing rule of the application has $n+1$ premisses. In particular, in the case where $n = 0$, we obtain the following rule: $\dfrac{\Gamma_0 \vdash_R v : ([], \alpha)}{\Gamma_0 \vdash_R (v)u : \alpha}$ for any λ-term u. So, the empty multiset plays the role of the universal type Ω.

The intersection we consider is *not* idempotent in the following sense: if a closed λ-term t has the type $a_1 \dots a_m \alpha$ and, for $1 \leq j \leq m$, $\mathsf{Supp}(a'_j) = \mathsf{Supp}(a_j)$, it does not follow necessarily that t has the type $a'_1 \dots a'_m \alpha$. For instance, the λ-term $\lambda z.\lambda x.(z)x$ has types $([([\alpha], \alpha)], ([\alpha], \alpha))$ and $([([\alpha, \alpha], \alpha)], ([\alpha, \alpha], \alpha))$ but not the type $([([\alpha], \alpha)], ([\alpha, \alpha], \alpha))$. On the contrary, the system presented in [Ronchi Della Rocca 1988] and the System $\mathcal{D}$ presented in [Krivine 1990] consider an idempotent intersection. System λ of [Kfoury 2000] and System $\mathbb{I}$ of [Neergaard and Mairson 2004] consider a non-idempotent intersection, but the treatment of weakening is not the same.

Interestingly, System R can be seen as a reformulation of the sytem of [Coppo *et al.* 1980]. More precisely, types of System R correspond to their normalized types. As stated in Section 5 of that paper, the authors thought that a particular property should hold in the corresponding semantics (assertion vi) of their Theorem 8. But our Proposition 16 shows that this is not the case.

3.2 Relating types and semantics

We prove in this subsection that the semantics of a closed λ-term as defined in Subsection 2.3 is the set of its types in System R. The following assertions relate more precisely types and semantics of any λ-term.

Theorem 20 *For any λ-term t such that $FV(t) \subseteq \{x_1, \ldots, x_m\}$, we have*

$$[\![t]\!]_{x_1,\ldots,x_m} = \{((a_1, \ldots, a_m), \alpha) \in (\prod_{j=1}^{m} \mathcal{M}_f(D)) \times D \;/\; x_1 : a_1, \ldots, x_m : a_m \vdash_R t : \alpha\} \;.$$

Proof. By induction on t. □

Corollary 21 *For any λ-terms t and t' such that $t =_\beta t'$, if $\Gamma \vdash_R t : \alpha$, then we have $\Gamma \vdash_R t' : \alpha$.*

Theorem 22 *For any λ-term t and for any $\Gamma \in \Phi$, we have*

$$\{\alpha \in D \;/\; \Gamma \vdash_R t : \alpha\} \subseteq \{\alpha \in D \;/\; \forall \rho \in \mathcal{P}(D)^{\mathcal{V}} \; (\forall x \in \mathcal{V} \; \Gamma(x) \in \mathcal{M}_f(\rho(x)) \Rightarrow \alpha \in [\![t]\!]_\rho)\}.$$

Proof. Apply Theorem 20. □

Remark 23 *The reverse inclusion is not true.*

Theorem 24 *For any λ-term t and for any $\rho \in \mathcal{P}(D)^{\mathcal{V}}$, we have*

$$[\![t]\!]_\rho = \{\alpha \in D \;/\; \exists \Gamma \in \Phi \; (\forall x \in \mathcal{V} \; \Gamma(x) \in \mathcal{M}_f(\rho(x)) \text{ and } \Gamma \vdash_R t : \alpha)\} \;.$$

Proof. Apply Theorems 20 and 22. □

There is another way to compute the interpretation of λ-terms in this semantics. Indeed, it is well-known that we can translate λ-terms into linear logic proof nets labelled with the types I, O, $?I$ and $!O$ (as in [Regnier 1992]): this translation is defined by induction on the λ-terms. Now, we can do experiments to compute the semantics of the proof net in the multiset based relational model: all the translations corresponding to the encoding $A \Rightarrow B \equiv ?A^\perp \wp B$ have the same semantics. And this semantics is the same as the semantics defined here.

For a survey of translations of λ-terms in proof nets, see [Guerrini 2004].

3.3 An equivalence relation on derivations

Definition 26 introduces an equivalence relation on the set of derivations of a given λ-term. This relation, as well as the notion of substitution defined immediately after, will play a role in Subsection 5.5.

Definition 25 *For any λ-term t, for any $(\Gamma, \alpha) \in \Phi \times D$, we denote by $\Delta(t, (\Gamma, \alpha))$ the set of derivations of $\Gamma \vdash_R t : \alpha$.*

For any closed λ-term t, for any $\alpha \in D$, we denote by $\Delta(t, \alpha)$ the set of derivations of $\vdash_R t : \alpha$.

For any λ-term t, we set $\Delta(t) = \bigcup_{(\Gamma,\alpha) \in \Phi \times D} \Delta(t, (\Gamma, \alpha))$.

Definition 26 *Let t be a λ-term t. For any $\Pi, \Pi' \in \Delta(t)$, we define, by induction on Π, when $\Pi \sim \Pi'$ holds:*

- *if Π is only a leaf, then $\Pi \sim \Pi'$ if, and only if, Π' is a leaf too;*
- *if* $\Pi = \dfrac{\begin{array}{c}\Pi_0\\ \Gamma, x : a \vdash_R v : \alpha\end{array}}{\Gamma \vdash_R \lambda x.v : (a, \alpha)}$ *, then $\Pi \sim \Pi'$ if, and only if,* $\Pi' = \dfrac{\begin{array}{c}\Pi'_0\\ \Gamma', x : a' \vdash_R v : \alpha'\end{array}}{\Gamma' \vdash_R \lambda x.v : (a', \alpha')}$ *and $\Pi_0 \sim \Pi'_0$;*
- *if*

$$\Pi = \dfrac{\begin{array}{c}\Pi_0\\ \Gamma_0 \vdash_R v : ([\alpha_1, \ldots, \alpha_n], \alpha)\end{array} \qquad \begin{array}{c}\Pi_1\\ \Gamma_1 \vdash_R u : \alpha_1\end{array} \quad \begin{array}{c}\ldots\\ \ldots\end{array} \quad \begin{array}{c}\Pi_n\\ \Gamma_n \vdash u : \alpha_n\end{array}}{\Gamma_0 + \Gamma_1 + \ldots + \Gamma_n \vdash_R (v)u : \alpha} \ ,$$

  then $\Pi \sim \Pi'$ if, and only if,

  – $\Pi' = \dfrac{\begin{array}{c}\Pi'_0\\ \Gamma'_0 \vdash_R v : ([\alpha'_1, \ldots, \alpha'_n], \alpha')\end{array} \qquad \begin{array}{c}\Pi'_1\\ \Gamma'_1 \vdash_R u : \alpha'_1\end{array} \quad \begin{array}{c}\ldots\\ \ldots\end{array} \quad \begin{array}{c}\Pi'_n\\ \Gamma'_n \vdash u : \alpha'_n\end{array}}{\Gamma'_0 + \Gamma'_1 + \ldots + \Gamma'_n \vdash_R (v)u : \alpha'} \ ,$,

  – $\Pi_0 \sim \Pi'_0$

  – *and there exists a permutation $\sigma \in \mathfrak{S}_n$ s.t., for any $i \in \{1, \ldots, n\}$, $\Pi_i \sim \Pi'_{\sigma(i)}$.*

An equivalence class of derivations of a λ-term t in System R can be seen as a *simple resource term of the shape of t* that does not reduce to 0. *Resource λ-calculus* is defined in [Ehrhard and Regnier 2006] and is similar to resource oriented versions of the λ-calculus previously introduced and studied in [Boudol *et al.* 1999] and [Kfoury 2000]. For a full exposition of a precise relation between this equivalence relation and simple resource terms, see [de Carvalho 2007].

Definition 27 *A substitution σ is a function from D to D such that*

$$\text{for any } \alpha, \alpha_1, \ldots, \alpha_n \in D,\ \sigma([\alpha_1, \ldots, \alpha_n], \alpha) = ([\sigma(\alpha_1), \ldots, \sigma(\alpha_n)], \sigma(\alpha)) \ .$$

We denote by $\mathcal{S}$ the set of substitutions.

For any $\sigma \in \mathcal{S}$, we denote by $\overline{\sigma}$ the function from $\mathcal{M}_f(D)$ to $\mathcal{M}_f(D)$ defined by $\overline{\sigma}([\alpha_1, \ldots, \alpha_n]) = [\sigma(\alpha_1), \ldots, \sigma(\alpha_n)]$.

Proposition 28 *Let Π be a derivation of $\Gamma \vdash_R t : \alpha$ and let σ be a substitution. Then there exists a derivation Π' of $\overline{\sigma} \circ \Gamma \vdash_R t : \sigma(\alpha)$ such that $\Pi \sim \Pi'$.*

Proof. By induction on t. □

4 Qualitative results

In this section, inspired by [Krivine 1990], we prove Theorem 33, which formulates *qualitative* relations between assignable types and normalization properties: it characterizes the (head) normalizable λ-terms by semantics means. We also answer to the following question: if v and u are two closed normal λ-terms, is it the case that $(v)u$ is (head) normalizable? The answer is given only referring to $[\![v]\!]$ and $[\![u]\!]$ in Corollary 34. Quantitative versions of this last result will be proved in Section 5.

Proposition 29 *(i) Every head-normalizable λ-term is typable in System R.*

(ii) For any normalizable λ-term t, there exist $\alpha \in D$ in which $[]$ has no positive occurrences and $\Gamma \in \Phi$ in which $[]$ has no negative occurrences such that $\Gamma \vdash_R t : \alpha$.

PROOF.

(i) Let t be a head-normalizable λ-term. There exist $k, n \in \mathbb{N}$, $x, x_1, \ldots, x_k \in \mathcal{V}$, n λ-terms $v_1, \ldots, v_n$ such that $(\lambda x_1 \ldots . \lambda x_k.t)v_1 \ldots v_n =_\beta x$. Now, x is typable. Therefore, by Corollary 21, the λ-term $(\lambda x_1 \ldots . \lambda x_k.t)v_1 \ldots v_n$ is typable. Hence $\lambda x_1 \ldots . \lambda x_k.t$ is typable.

(ii) We prove, by induction on t, that for any normal λ-term t, the following properties hold:

- there exist $\alpha \in D$ in which $[]$ has no positive occurrences and $\Gamma \in \Phi$ in which $[]$ has no negative occurrences such that $\Gamma \vdash_R t : \alpha$;
- if, moreover, t does not begin with λ, then, for any $\alpha \in D$ in which $[]$ has no positive occurrences, there exists $\Gamma \in \Phi$ in which $[]$ has no negative occurrences such that $\Gamma \vdash_R t : \alpha$.

Next, just apply Corollary 21.

$\square$

If $\mathcal{X}_1$ and $\mathcal{X}_2$ are two sets of λ-terms, then $\mathcal{X}_1 \to \mathcal{X}_2$ denotes the set of λ-terms v such that for any $u \in \mathcal{X}_1$, $(v)u \in \mathcal{X}_2$. A set $\mathcal{X}$ of λ-terms is said to be *saturated* if for any λ-terms $t_1, \ldots, t_n, u$ and for any $x \in \mathcal{V}$, $((u[t/x])t_1 \ldots t_n \in \mathcal{X} \Rightarrow (\lambda x.u)tt_1 \ldots t_n \in \mathcal{X})$. An interpretation is a map from A to the set of saturated set. For any interpretation $\mathcal{I}$ and for any $\delta \in D \cup \mathcal{M}_f(D)$, we define, by induction on δ, a saturated set $|\delta|_\mathcal{I}$:

- if $\delta \in A$, then $|\delta|_\mathcal{I} = \mathcal{I}(\delta)$;
- if $\delta = []$, then $|\delta|_\mathcal{I}$ is the set of all λ-terms ;
- if $\delta = [\alpha_1, \ldots, \alpha_{n+1}]$, then $|\delta|_\mathcal{I} = \bigcap_{i=1}^{n+1} |\alpha_i|_\mathcal{I}$.
- if $\delta = (a, \alpha)$, then $|\delta|_\mathcal{I} = |a|_\mathcal{I} \to |\alpha|_\mathcal{I}$.

Lemma 30 *Let $\mathcal{I}$ be an interpretation and let u be a λ-term such that $x_1 : a_1, \ldots, x_k : a_k \vdash_R u : \alpha$. If $t_1 \in |a_1|_\mathcal{I}, \ldots, t_k \in |a_k|_\mathcal{I}$, then $u[t_1/x_1, \ldots, t_k/x_k] \in |\alpha|_\mathcal{I}$.*

PROOF. By induction on u. $\square$

Lemma 31 *(i) Let $\mathcal{N}$ be the set of head-normalizable terms. For any $\gamma \in A$, we set $\mathcal{I}(\gamma) = \mathcal{N}$. Then, for any $\alpha \in D$, we have $\mathcal{V} \subseteq |\alpha|_\mathcal{I} \subseteq \mathcal{N}$.*

(ii) Let $\mathcal{N}$ be the set of normalizable terms. For any $\gamma \in A$, we set $\mathcal{I}(\gamma) = \mathcal{N}$. For any $\alpha \in D$ with no negative (respectively positive) occurences of $[]$, we have $\mathcal{V} \subseteq |\alpha|_\mathcal{I}$ (respectively $|\alpha|_\mathcal{I} \subseteq \mathcal{N}$).

PROOF.

(i) Set $\mathcal{N}_0 = \{(x)t_1 \dots t_n \ / \ x \in \mathcal{V}, t_1, \dots, t_n \in \Lambda\}$. We prove, by induction on α, that we have $\mathcal{N}_0 \subseteq |\alpha|_\mathcal{I} \subseteq \mathcal{N}$.

If $\alpha = (b, \beta)$, then, by induction hypothesis, we have $\mathcal{N}_0 \subseteq |\beta|_\mathcal{I} \subseteq \mathcal{N}$ and $\mathcal{N}_0 \subseteq |b|_\mathcal{I}$. Hence we have $\mathcal{N}_0 \subseteq \Lambda \rightarrow \mathcal{N}_0 \subseteq |\alpha|_\mathcal{I}$ and $|\alpha|_\mathcal{I} \subseteq \mathcal{N}_0 \rightarrow \mathcal{N} \subseteq \mathcal{N}$.

(ii) Set $\mathcal{N}_0 = \{(x)t_1 \dots t_n \ / \ x \in \mathcal{V}, t_1, \dots, t_n \in \mathcal{N}\}$. We prove, by induction on α, that

- if $[]$ has no negative occurrences in α, then we have $\mathcal{N}_0 \subseteq |\alpha|_\mathcal{I}$;
- if $[]$ has no positive occurrences in α, then we have $|\alpha|_\mathcal{I} \subseteq \mathcal{N}$.

Suppose $\alpha = (b, \beta) \in \mathcal{M}_f(D) \times D$.

- If $[]$ has no negative occurrences in α, then $[]$ has no positive occurrences (respectively negative) in b (respectively in β). By induction hypothesis, we have $|b|_\mathcal{I} \subseteq \mathcal{N}$ and $\mathcal{N}_0 \subseteq |\beta|_\mathcal{I}$. Hence $\mathcal{N}_0 \subseteq \mathcal{N} \rightarrow \mathcal{N}_0 \subseteq |b|_\mathcal{I} \rightarrow |\beta|_\mathcal{I} = |\alpha|_\mathcal{I}$.
- If $[]$ has no positive occurrences in α, then $[]$ has no negative occurrences (respectively positive) in b (respectively in β). By induction hypothesis, we have $\mathcal{N}_0 \subseteq |b|_\mathcal{I}$ and $|\beta|_\mathcal{I} \subseteq \mathcal{N}$. Donc $|\alpha|_\mathcal{I} = |b|_\mathcal{I} \rightarrow |\beta|_\mathcal{I} \subseteq \mathcal{N}_0 \rightarrow \mathcal{N} \subseteq \mathcal{N}$ (this last inclusion follows from the fact that for any λ-term t, for any variable x that is not free in t, if $(t)x$ is normalizable, then t is normalizable, fact that can be proved by induction on the number of left-reductions of $(t)x$).

□

Proposition 32 *(i) Every typable λ-term in System R is head-normalizable.*

(ii) Let t be a λ-term, $\alpha \in D$ in which $[]$ has no positive occurrences and $\Gamma \in \Phi$ in which $[]$ has no negative occurrences such that $\Gamma \vdash_R t : \alpha$. Then t is normalizable.

PROOF.

(i) Let Γ be the context $x_1 : a_1, \dots, x_k : a_k$. For any $\gamma \in A$, we set $\mathcal{I}(\gamma) = \mathcal{N}$, where $\mathcal{N}$ is the set of head-normalizable terms. By Lemma 31 (i), we have $x_1 \in |a_1|_\mathcal{I}, \dots, x_k \in |a_k|_\mathcal{I}$. Hence, by Lemma 30, we have $t = t[x_1/x_1, \dots, x_k/x_k] \in |\alpha|_\mathcal{I}$. Using again Lemma 31 (i), we obtain $|\alpha|_\mathcal{I} \subseteq \mathcal{N}$.

(ii) Let Γ be the context $x_1 : a_1, \dots, x_k : a_k$. For any $\gamma \in A$, we set $\mathcal{I}(\gamma) = \mathcal{N}$, where $\mathcal{N}$ is the set of normalizable terms. By Lemma 31 (ii), we have $x_1 \in |a_1|_\mathcal{I}, \dots, x_k \in |a_k|_\mathcal{I}$. Hence, by Lemma 30, we have $t = t[x_1/x_1, \dots, x_k/x_k] \in |\alpha|_\mathcal{I}$. Using again Lemma 31 (ii), we obtain $|\alpha|_\mathcal{I} \subseteq \mathcal{N}$.

□

Theorem 33 *(i) For any $t \in \Lambda$, t is head-normalizable if, and only if, t is typable in System R.*

(ii) For any $t \in \Lambda$, t is normalizable if, and only if, there exist $\alpha \in D$ in which $[]$ has no positive occurrences and $\Gamma \in \Phi$ in which $[]$ has no negative occurrences such that $\Gamma \vdash_R t : \alpha$.

PROOF.

(i) Apply Proposition 29 (i) and Proposition 32 (i).

(ii) Apply Proposition 29 (ii) and Proposition 32 (ii).

□

This theorem is not surprising: although System R is not considered in [Dezani-Ciancaglini *et al.*], it is quite obvious that its typing power is the same as that of the systems containing Ω considered by this paper. We can note here a difference with Systems λ and $\mathbb{I}$ already mentioned: in these systems, only strongly normalizable terms are typable.

Corollary 34 *Let v and u two closed normal terms.*

(i) There exist $a \in \mathcal{M}_f(D)$, $\alpha \in D$ such that $(a, \alpha) \in [\![v]\!]$ and $\mathsf{Supp}(a) \subseteq [\![u]\!]$ if, and only if, $(v)u$ is head-normalizable.

(ii) There exist $a \in \mathcal{M}_f(D)$, $\alpha \in D$ such that $(a, \alpha) \in [\![v]\!]$, $\mathsf{Supp}(a) \subseteq [\![u]\!]$ and $[]$ has no positive occurrences in α if, and only if, $(v)u$ is normalizable.

5 Quantitative results

We now turn our attention to the quantitative aspects of reduction. The aim is to give a purely semantic account of execution time. Of course, if t' is the normal form of t, we know that $[\![t]\!] = [\![t']\!]$, so that from $[\![t]\!]$ it is clearly impossible to determine the number of reduction steps from t to t'. Nevertheless, if v and u are two normal λ-terms, we can wonder what is the number of steps leading from $(v)u$ to its (principal head) normal norm. We prove in this section that we can answer the question by only referring to $[\![v]\!]$ and $[\![u]\!]$ (Theorem 59).

5.1 Type Derivations for States

We now extend the type derivations for λ-terms to type derivations for closures (Definition 35) and for states (Definition 38). We will define also the size $|\Pi|$ of such derivations Π; naturally, the size $|\Pi|$ of a derivation Π of System R is quite simply its size as a tree, i.e. the number of its nodes.

Definition 35 *For any closure $c = (t, e)$, for any $(\Gamma, \alpha) \in \Phi \times D$ (respectively $(\Gamma, a) \in \Phi \times \mathcal{M}_f(D)$), we define, by induction on $d(e)$, what is a derivation Π of $\Gamma \vdash c : \alpha$ (respectively $\Gamma \vdash c : a$) and what is $|\Pi|$ for such a derivation :*

- – *if $e = \emptyset$, then a derivation of $\Gamma \vdash c : \alpha$ is a pair $(\Pi_0, \emptyset)$ with $\Pi_0 \in \Delta(t, (\Gamma, \alpha))$;*
  – *if $e = \{(x_1, c_1), \ldots, (x_m, c_m)\}$ with $m \neq 0$, then a derivation of $\Gamma \vdash c : \alpha$ is a pair $(\Pi_0, \{(x_1, \Pi_1), \ldots, (x_m, \Pi_m)\})$, where*
    - ∗ *Π_0 is a derivation of $\Gamma_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R t : \alpha$;*
    - ∗ *for any $j \in \{1, \ldots, m\}$, Π_j is a derivation of $\Gamma_j \vdash c_j : a_j$;*
    - ∗ *and $\Gamma = \sum_{j=0}^{m} \Gamma_j$.*

  If $\Pi = (\Pi_0, \{(x_1, \Pi_1), \ldots, (x_m, \Pi_m)\})$ is a derivation of $\Gamma \vdash c : \alpha$, then we set $|\Pi| = \sum_{j=0}^{m} |\Pi_j|$.

- *For any integer* p*, a derivation of* $\Gamma \vdash c : [\alpha_1, \ldots, \alpha_p]$ *is a* p*-tuple* $(\Pi^1, \ldots, \Pi^p)$ *such that there exists* $(\Gamma^1, \ldots, \Gamma^p) \in \Phi^p$ *and*
  - *for* $1 \leq i \leq p$*,* Π^i *is a derivation of* $\Gamma^i \vdash c : \alpha_i$*;*
  - *and* $\Gamma = \sum_{i=1}^{p} \Gamma^i$*;*

  If $\Pi = (\Pi^1, \ldots, \Pi^p)$ *is a derivation of* $\Gamma \vdash c : a$*, then we set* $|\Pi| = \sum_{i=1}^{p} |\Pi^i|$.

Definition 35 is not so easy to use directly. This is why we introduce Lemmas 36 and 37, that will be useful for proving Propositions 43 and 49.

Lemma 36 *Let* $((v)u, e) \in \mathcal{C}$*. For any* $b \in \mathcal{M}_f(D)$*,* $\Gamma', \Gamma'' \in \Phi$*, if* Π' *is a derivation of* $\Gamma' \vdash (v, e) : (b, \alpha)$ *and* Π'' *is a derivation of* $\Gamma'' \vdash (u, e) : b$*, then there exists a derivation* Π *of* $\Gamma' + \Gamma'' \vdash ((v)u, e) : \alpha$ *such that* $|\Pi| = |\Pi'| + |\Pi''| + 1$*.*

Proof. Set $e = \{(x_1, c_1), \ldots, (x_m, c_m)\}$ and $\Pi' = (\Pi'_0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\})$, where

- Π'_0 is a derivation of $\Gamma'_0, x_1 : a'_1, \ldots, x_m : a'_m \vdash_R v : (b, \alpha)$,
- for $1 \leq j \leq m$, Π'_j is a derivation of $\Gamma'_j \vdash c_j : a'_j$
- and $\Gamma' = \sum_{j=0}^{m} \Gamma'_j$.

Set $b = [\beta_1, \ldots, \beta_p]$ and $\Pi'' = (\Pi''^1, \ldots, \Pi''^p)$ where, for $1 \leq k \leq p$, Π''^k is a derivation $(\Pi''^k_0, \{(x_j, \Pi''^k_j)\}_{1 \leq j \leq m})$ of $\Gamma''^k \vdash (u, e) : \beta_k$ with $\Gamma'' = \sum_{k=1}^{p} \Gamma''^k$. For $k \in \{1, \ldots, p\}$,

- Π''^k_0 is a derivation of $\Gamma''^k_0, x_1 : a''^k_1, \ldots, x_m : a''^k_m \vdash_R u : \beta_k$,
- for $1 \leq j \leq m$, Π''^k_j is a derivation of $\Gamma''^k_j \vdash c_j : a''^k_j$
- and $\Gamma''^k = \sum_{j=0}^{m} \Gamma''^k_j$.

For $j \in \{0, \ldots, m\}$, we set $\Gamma_j = \Gamma'_j + \sum_{k=1}^{p} \Gamma''^k_j$ and $a_j = a'_j + \sum_{k=1}^{p} a''^k_j$. There exists a derivation Π_0 of $\Gamma_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R (v)u : \alpha$ with $|\Pi_0| = |\Pi'_0| + \sum_{k=1}^{p} |\Pi''^k_0| + 1$. Moreover, for $j \in \{1, \ldots, m\}$, $\Pi_j = \Pi'_j * \Pi''^1_j * \ldots * \Pi''^p_j$, where $*$ is the concatenation of finite sequences, is a derivation of $\Gamma_j \vdash c_j : a_j$. We have

$$\begin{aligned} \sum_{j=1}^{m} \Gamma_j &= \sum_{j=0}^{m} (\Gamma'_j + \sum_{k=1}^{p} \Gamma''^k_j) \\ &= \sum_{j=0}^{m} \Gamma'_j + \sum_{j=0}^{m} \sum_{k=1}^{p} \Gamma''^k_j \\ &= \Gamma' + \Gamma'' \ . \end{aligned}$$

Hence $\Pi = (\Pi_0, \{(x_1, \Pi_1), \ldots, (x_m, \Pi_m)\})$ is a derivation of $\Gamma' + \Gamma'' \vdash ((v)u, e) : \alpha$. We have

$$\begin{aligned} |\Pi| &= \sum_{j=0}^{m} |\Pi_j| \\ &= |\Pi'_0| + \sum_{k=1}^{p} |\Pi''^k_0| + 1 + \sum_{j=1}^{m} |\Pi_j| \end{aligned}$$

$$\begin{aligned}
&= |\Pi'_0| + \sum_{k=1}^{p} |\Pi''^k_0| + 1 + \sum_{j=1}^{m} (|\Pi'_j| + \sum_{k=1}^{p} |\Pi''^k_j|) \\
&= |\Pi'| + \sum_{k=1}^{p} |\Pi''^k| + 1 \\
&= |\Pi'| + |\Pi''| + 1 \ .
\end{aligned}$$

□

Lemma 37 *For any closure (u, e), for any derivation Π' of $\Gamma, x : b \vdash (u, e) : \beta$, there exists a derivation Π of $\Gamma \vdash (\lambda x.u, e) : (b, \beta)$ such that $|\Pi| = |\Pi'| + 1$.*

PROOF. We set $e = \{(x_1, c_1), \ldots, (x_m, c_m)\}$ and $\Pi' = (\Pi'_0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\})$. We know that Π'_0 is a derivation of $\Gamma, x : b, x_1 : a_1, \ldots, x_m : a_m \vdash_R u : \beta$, hence there exists a derivation Π_0 of $\Gamma, x_1 : a_1, \ldots, x_m : a_m \vdash_R \lambda x.u : (b, \beta)$. We set $\Pi = (\Pi_0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\})$: it is a derivation of $\Gamma \vdash (\lambda x.u, e) : (b, \beta)$ and we have

$$\begin{aligned}
|\Pi| &= |\Pi_0| + \sum_{j=1}^{m} |\Pi'_j| \\
&= |\Pi'_0| + 1 + \sum_{j=1}^{m} |\Pi'_j| \\
&= |\Pi'| + 1 \ .
\end{aligned}$$

□

Definition 38 *Let $s = (c, (c_1, \ldots, c_q))$ be a state. A pair $(\Pi_0, (\Pi_1, \ldots, \Pi_q))$ is said to be a derivation of $\Gamma \vdash s : \alpha$ if there exist $b_1, \ldots, b_q \in \mathcal{M}_f(D)$, $\Gamma_0, \ldots, \Gamma_q \in \Phi$ such that*

- *Π_0 is a derivation of $\Gamma_0 \vdash c : b_1 \ldots b_q \alpha$;*
- *for any $k \in \{1, \ldots, q\}$, Π_k is a derivation of $\Gamma_k \vdash c_k : b_k$;*
- *and $\Gamma = \sum_{k=0}^{q} \Gamma_k$.*

In this case, we set $|(\Pi_0, (\Pi_1, \ldots, \Pi_q))| = \sum_{k=0}^{q} |\Pi_k|$.

Definition 38 is not so easy to use directly. This is why we introduce Lemmas 39 and 40, that will be useful for proving Propositions 43 and 49.

Lemma 39 *Let $m, j_0 \in \mathbb{N}$ such that $1 \leq j_0 \leq m$. Let $s = (c'_{j_0}, (c_1, \ldots, c_q)) \in \mathbb{S}$, $x_1, \ldots, x_m \in \mathcal{V}$, $c'_1, \ldots, c'_m \in \mathcal{C}$. For any $(\Gamma, \alpha) \in \Phi \times D$, if Π' is a derivation of $\Gamma \vdash s : \alpha$, then there exists a derivation Π of $\Gamma \vdash ((x_{j_0}, \{(x_1, c'_1), \ldots, (x_m, c'_m)\}), (c_1, \ldots, c_q)) : \alpha$ such that $|\Pi| = |\Pi'| + 1$.*

PROOF. We set $\Pi' = (\Pi'', (\Pi_1, \ldots, \Pi_q))$ with Π'' a derivation of $\Gamma'' \vdash c'_{j_0} : b_1 \ldots b_q \alpha$. We denote by Π_0 the derivation of $x : [b_1 \ldots b_q \alpha] \vdash_R x : b_1 \ldots b_q \alpha$. For any $j \in \{1, \ldots, m\}$, we set

$$\Pi''_j = \begin{cases} (\Pi'') & \text{if } j = j_0 \ ; \\ \epsilon & \text{else.} \end{cases}$$

The pair $((\Pi_0, \{(x_1, \Pi''_1), \ldots, (x_m, \Pi''_m)\}), (\Pi_1, \ldots, \Pi_q))$ is a derivation of

$$\Gamma \vdash ((x_{j_0}, \{(x_1, c'_1), \ldots, (x_m, c'_m)\}), (c_1, \ldots, c_q)) : \alpha$$

and we have

$$\begin{aligned} |((\Pi_0, \{(x_1, \Pi''_1), \ldots, (x_m, \Pi''_m)\}, (\Pi_1, \ldots, \Pi_q))| &= \sum_{i=1}^{n} |\Pi''_1| + \sum_{k=1}^{q} |\Pi_k| + |\Pi_0| \\ &= |\Pi''| + \sum_{k=1}^{q} |\Pi_k| + 1 \\ &= |(\Pi'', (\Pi_1, \ldots, \Pi_q))| + 1 \\ &= |\Pi'| + 1 \ . \end{aligned}$$

□

Lemma 40 *For any state $s = ((u, \{(x, c)\} \cup e), (c_1, \ldots, c_q))$, for any derivation Π' of $\Gamma \vdash s : \alpha$, there exists a derivation Π of $\Gamma \vdash ((\lambda x.u, e), (c, c_1, \ldots, c_q)) : \alpha$ s.t. $|\Pi| = |\Pi'| + 1$.*

Proof. Set $e = \{(x_1, c'_1), \ldots, (x_m, c'_m)\}$ and

$$\Pi' = ((\Pi'_0, \{(x, \Pi''), (x_1, \Pi''_1), \ldots, (x_m \Pi''_m)\}), (\Pi'_1, \ldots, \Pi'_q)) \ .$$

We know that Π'_0 is a derivation of $\Gamma, x_1 : a_1, \ldots, x_m : a_m, x : a \vdash_R u : b_1 \ldots b_q\alpha$, hence there exists a derivation Π_0 of $x_1 : a_1, \ldots, x_m : a_m \vdash_R \lambda x.u : ab_1 \ldots b_q\alpha$ such that $|\Pi_0| = |\Pi'_0| + 1$. Set $\Pi = ((\Pi_0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\}), (\Pi'', \Pi'_1, \ldots, \Pi'_q))$: it is a derivation of $\Gamma \vdash ((\lambda x.u, e), (c, c_1, \ldots, c_q)) : \alpha$ and we have

$$\begin{aligned} |\Pi| &= |\Pi_0| + \sum_{j=1}^{m} |\Pi'_j| + |\Pi''| + \sum_{k=1}^{q} |\Pi'_k| \\ &= |\Pi'_0| + 1 + \sum_{j=1}^{m} |\Pi'_j| + |\Pi''| + \sum_{k=1}^{q} |\Pi'_k| \\ &= |\Pi'| + 1 \ . \end{aligned}$$

□

5.2 Relating size of derivations and execution time

The aim of this subsection is to prove Theorem 44, that gives the exact number of steps leading to the principal head normal form by means of derivations in System R.

Lemma 41 *Let $((v)u, e)$ be a closure and let $(\Gamma, \alpha) \in \Phi \times D$. For any derivation Π of $\Gamma \vdash ((v)u, e) : \alpha$, there exist $b \in \mathcal{M}_f(D)$, $\Gamma', \Gamma'' \in \Phi$, a derivation Π' of $\Gamma' \vdash (v, e) : (b, \alpha)$ and a derivation Π'' of $\Gamma'' \vdash (u, e) : b$ such that $\Gamma = \Gamma' + \Gamma''$ and $|\Pi| = |\Pi'| + |\Pi''| + 1$.*

Proof. Set $e = \{(x_1, c_1), \ldots, (x_m, c_m)\}$ and $\Pi = (\Pi_0, \{(x_1, \Pi_1), \ldots, (x_m, \Pi_m)\})$ where

(i) Π_0 is a derivation of $\Gamma_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R (v)u : \alpha$,

(ii) for $1 \leq j \leq m$, Π_j is a derivation of $\Gamma_j \vdash c_j : a_j$,

(iii) $\Gamma = \sum_{j=0}^{m} \Gamma_j$.

By (i), there exist $p \in \mathbb{N}$, $\beta_1, \ldots, \beta_p \in D$, a derivation Π_0^0 of $\Gamma_0^0, x_1 : a'_1, \ldots, x_m : a'_m \vdash_R v : ([\beta_1, \ldots, \beta_p], \alpha)$ and, for $1 \leq k \leq p$, a derivation Π_0^k of $\Gamma_0^k, x_1 : {a''_1}^k, \ldots, x_m : {a''_m}^k \vdash_R u : \beta_k$ such that

- $\Gamma_0 = \sum_{k=0}^{p} \Gamma_0^k$,
- for $1 \leq j \leq m$, $a_j = a'_j + \sum_{k=1}^{p} {a''_j}^k$
- and $|\Pi_0| = \sum_{k=0}^{p} |\Pi_0^k| + 1$.

For any $j \in \{1, \ldots, m\}$, we set $a''_j = \sum_{k=1}^{p} {a''_j}^k$. By (ii), for any $j \in \{1, \ldots, m\}$, there exist Γ'_j, Γ''_j, a derivation Π'_j of $\Gamma'_j \vdash c_j : a'_j$ and a derivation Π''_j of $\Gamma''_j \vdash c_j : a''_j$ such that

- $\Gamma_j = \Gamma'_j + \Gamma''_j$
- and $|\Pi_j| = |\Pi'_j| + |\Pi''_j|$.

Set

- $b = [\beta_1, \ldots, \beta_p]$,
- $\Gamma' = \Gamma_0^0 + \sum_{j=1}^{m} \Gamma'_j$,
- $\Gamma'' = \sum_{k=1}^{p} \Gamma_0^k + \sum_{j=1}^{m} \Gamma''_j$,
- $\Pi' = (\Pi_0^0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\})$
- and $\Pi'' = ((\Pi_0^1, \ldots, \Pi_0^p), \{(x_1, \Pi''_1), \ldots, (x_m, \Pi''_m)\})$.

We have

$$\begin{array}{rcl} \Gamma & = & \displaystyle\sum_{j=0}^{m} \Gamma_m \\ & & \text{(by (iii))} \\ & = & \displaystyle\sum_{k=0}^{p} \Gamma_0^k + \sum_{j=1}^{m} (\Gamma'_j + \Gamma''_j) \\ & = & \Gamma' + \Gamma'' \end{array}$$

and

$$\begin{array}{rcl} |\Pi| & = & \displaystyle\sum_{j=0}^{m} |\Pi_j| \\ & = & \displaystyle\sum_{k=0}^{p} |\Pi_0^k| + 1 + \sum_{j=1}^{m} (|\Pi'_j| + |\Pi''_j|) \\ & = & |\Pi'| + |\Pi''| + 1 \ . \end{array}$$

□

Proposition 42 *Let t be a head normalizable λ-term. For any $(\Gamma, \alpha) \in \Phi \times D$, for any $\Pi \in \Delta(t, (\Gamma, \alpha))$, we have $l_h((t, \emptyset), \epsilon) \leq |\Pi|$.*

PROOF. By Theorem 10, we can prove, by induction on $l_h(s)$, that for any $s \in \mathbb{S}$ such that $\overline{s}$ is head normalizable, for any $(\Gamma, \alpha) \in \Phi \times D$, for any derivation Π of $\Gamma \vdash s : \alpha$, we have $l_h(s) \leq |\Pi|$.

The base case is trivial, because we never have $l_h(s) = 0$. The inductive step is divided into five cases:

- In the case where $s = ((x, e), \pi)$, $x \in \mathcal{V}$ and $x \notin \mathrm{dom}(e)$, $l_h(s) = 1 \leq |\Pi|$.
- In the case where $s = ((x_{j_0}, \{(x_1, c'_1), \ldots, (x_m, c'_m)\}), (c_1, \ldots, c_q))$ and $1 \leq j_0 \leq m$, we have $\Pi = (\Pi_0, (\Pi_1, \ldots, \Pi_q))$, where $\Pi_0 = (\Pi'_0, \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m)\})$ with
  - Π'_0 is a derivation of $\Gamma'_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R x_{j_0} : b_1 \ldots b_q \alpha$,
  - for any $j \in \{1, \ldots, m\}$, Π'_j is a derivation of $\Gamma'_j \vdash c'_j : a_j$,
  - $\Gamma_0 = \sum_{j=1}^{m} \Gamma'_j$,
  - for $1 \leq k \leq q$, Π_k is a derivation of $\Gamma_k \vdash c_k : b_k$
  - and $\Gamma = \sum_{k=0}^{q} \Gamma_k$.

  Hence $a'_{j_0} = [b_1 \ldots b_q \alpha]$. The pair $(\Pi'_{j_0}, (\Pi_1, \ldots, \Pi_q))$ is a derivation of
  $$\Gamma'_{j_0} + \sum_{k=1}^{q} \Gamma_k \vdash (c'_{j_0}, (c_1, \ldots, c_q)) : \alpha \ .$$
  We have
  $$\begin{array}{rcl} l_h(s) & = & l_h(c'_{j_0}, (c_1, \ldots, c_q)) + 1 \\ & \leq & |(\Pi'_{j_0}, (\Pi_1, \ldots, \Pi_q))| + 1 \\ & & \text{(by induction hypothesis)} \\ & = & |\Pi'_{j_0}| + \sum_{k=1}^{q} |\Pi_k| + 1 \\ & \leq & |\Pi_0| + \sum_{k=1}^{q} |\Pi_k| \\ & = & |\Pi| \ . \end{array}$$
- In the case where $s = ((\lambda x.u, \{(x_1, c'_1), \ldots, (x_n, c'_m)\}), (c', c_1, \ldots, c_q))$, we have $\Pi = ((\Pi'_0, \Pi''_0), (\Pi', \Pi_1, \ldots, \Pi_q))$ with
  - Π'_0 is a derivation of $\Gamma'_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R \lambda x.u : b' b_1 \ldots b_q \alpha$;
  - $\Pi''_0 = \{(x_j, \Pi'_j)\}_{1 \leq j \leq m}$ where, for $1 \leq j \leq m$, Π'_j is a derivation of $\Gamma'_j \vdash c'_j : a_j$;
  - $\Gamma_0 = \sum_{j=0}^{m} \Gamma'_j$;
  - Π' is a derivation of $\Gamma' \vdash b' : c'$;
  - for $1 \leq k \leq q$, Π_k is a derivation of $\Gamma_k \vdash b_k : c_k$.

Hence there exists a derivation Π'' of

$$\Gamma'_0, x_1 : a_1, \ldots, x_m : a_m, x : b' \vdash_R u : b_1 \ldots b_q\alpha$$

with $|\Pi'_0| = |\Pi''| + 1$. The pair $(\Pi'', \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m), \Pi'\})$ is a derivation of

$$\Gamma_0 + \Gamma' \vdash (u, \{(x_1, c'_1), \ldots, (x_m, c'_m), (x, c)\}) : b_1 \ldots b_q\alpha \ .$$

Hence $((\Pi'', \{(x_1, \Pi'_1), \ldots, (x_m, \Pi'_m), (x, \Pi')\}), (\Pi_1, \ldots, \Pi_q))$ is a derivation of

$$\Gamma \vdash ((u, \{(x_1, c'_1), \ldots, (x_m, c'_m), (x, c)\}), (c_1, \ldots, c_q)) : \alpha \ .$$

We have

$$\begin{array}{rcl} l_h(s) & = & l_h((u, \{(x_1, c'_1), \ldots, (x_m, c'_m), (x, c)\}), (c_1, \ldots, c_q)) + 1 \\ & \leq & |((\Pi'', \{\Pi'_1, \ldots, \Pi'_m, \Pi'\}), (\Pi_1, \ldots, \Pi_q))| + 1 \\ & & \text{(by induction hypothesis)} \\ & = & |\Pi''| + \sum_{j=1}^{m} |\Pi'_j| + |\Pi'| + \sum_{k=1}^{q} |\Pi_k| + 1 \\ & = & |\Pi'_0| + \sum_{j=1}^{m} |\Pi'_j| + |\Pi'| + \sum_{k=1}^{q} |\Pi_k| \\ & = & |\Pi'_0| + |\Pi''_0| + |\Pi'| + \sum_{k=1}^{q} |\Pi_k| \\ & = & |((\Pi'_0, \Pi''_0), (\Pi', \Pi_1, \ldots, \Pi_q))| \\ & = & |\Pi| \ . \end{array}$$

- In the case where $s = (((v)u, e), (c_1, \ldots, c_q))$, we have $\Pi = (\Pi_0, (\Pi_1, \ldots, \Pi_q))$ with
  - Π_0 is a derivation of $\Gamma_0 \vdash ((v)u, e) : b_1 \ldots b_q\alpha$;
  - for $1 \leq k \leq q$, Π_k is a derivation of $\Gamma_k \vdash c_k : b_k$;
  - $\Gamma = \sum_{k=0}^{q} \Gamma_k$.

  By Lemma 41, there exist $b \in \mathcal{M}_f(D)$, $\Gamma'_0, \Gamma''_0 \in \Phi$, a derivation Π'_0 of $\Gamma'_0 \vdash (u, e) : bb_1 \ldots b_q\alpha$ and a derivation Π''_0 of $\Gamma''_0 \vdash (u, e) : b$ such that $\Gamma_0 = \Gamma'_0 + \Gamma''_0$ and $|\Pi_0| = |\Pi'_0| + |\Pi''_0| + 1$. The pair $(\Pi'_0, (\Pi''_0, \Pi_1, \ldots, \Pi_q))$ is a derivation of $\Gamma \vdash ((v, e), ((u, e), c_1, \ldots, c_q)) : \alpha$. We have

$$\begin{array}{rcl} l_h(s) & = & l_h((v, e), ((u, e), c_1, \ldots, c_q)) + 1 \\ & \leq & |(\Pi'_0, (\Pi''_0, \Pi_1, \ldots, \Pi_q))| + 1 \\ & & \text{(by induction hypothesis)} \\ & = & |\Pi'_0| + |\Pi''_0| + \sum_{k=1}^{q} |\Pi_k| + 1 \\ & = & |\Pi_0| + \sum_{k=1}^{q} |\Pi_k| \\ & = & |(\Pi_0, (\Pi_1, \ldots, \Pi_q))| \\ & = & |\Pi| \ . \end{array}$$

- In the case where $s = ((\lambda x.u, \{(x_j, c'_j)\}_{1\leq j\leq m}), \epsilon)$, we have $\Pi = ((\Pi'_0, \Pi''_0), \epsilon)$ with
  - Π'_0 is a derivation of $\Gamma'_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R \lambda x.u : \alpha$;
  - $\Pi''_0 = \{(x_j, \Pi'_j)\}_{1\leq j\leq m}$ where, for $1 \leq j \leq m$, Π'_j is a derivation of $\Gamma'_j \vdash c'_j : a_j$;
  - $\Gamma = \sum_{j=0}^{m} \Gamma'_j$.

  Hence there exists a derivation Π'' of $\Gamma'_0, x_1 : a_1, \ldots, x_m : a_m, x : b \vdash_R u : \beta$ such that $\alpha = (b, \beta)$ and $|\Pi'_0| = |\Pi''| + 1$. The pair $((\Pi'', \Pi''_0), \epsilon)$ is a derivation of

$$\Gamma, x : b \vdash ((u, \{(x_1, c_1), \ldots, (x_m, c_m)\}), \epsilon) : \beta \ .$$

  We have

$$\begin{array}{rcl} l_h(s) & = & l_h((u, \{(x_1, c_1), \ldots, (x_m, c_m)\}), \epsilon) + 1 \\ & \leq & |((\Pi'', \Pi''_0), \epsilon)| + 1 \\ & = & |\Pi''| + \sum_{j=1}^{m} |\Pi'_j| + 1 \\ & = & |\Pi'_0| + \sum_{j=1}^{m} |\Pi'_j| \\ & = & |(\Pi'_0, \Pi''_0)| \\ & = & |\Pi| \ . \end{array}$$

□

Proposition 43 *Let t be a head normalizable λ-term. There exist $(\Gamma, \alpha) \in \Phi \times D$ and $\Pi \in \Delta(t, (\Gamma, \alpha))$ such that $l_h((t, \emptyset), \epsilon) = |\Pi|$.*

Proof. By Theorem 10, we can prove, by induction on $l_h(s)$, that for any $s \in \mathbb{S}$ such that $\overline{s}$ is head normalizable, there exist (Γ, α) and a derivation Π of $\Gamma \vdash s : \alpha$ such that we have $l_h(s) = |\Pi|$.

The base case is trivial, because we never have $l_h(s) = 0$. The inductive step is divided into five cases :

- In the case where $s = ((x, e), (c_1, \ldots, c_q))$, $x \in \mathcal{V}$ and $x \notin \mathsf{dom}(e)$, we have $l_h(s) = 1$ and there exists a derivation $\Pi = (\Pi_0, (\Pi_1, \ldots, \Pi_q))$ of $x : [\underbrace{[] \ldots []}_{q \text{ times}} \alpha] \vdash (x, e) : \underbrace{[] \ldots []}_{q \text{ times}} \alpha$ with $|\Pi_0| = 1$ and $|\Pi_1| = \ldots = |\Pi_q| = 0$.
- In the case where $s = ((x_{j_0}, \{(x_1, c'_1), \ldots, (x_m, c'_m)\}), (c_1, \ldots, c_q))$, apply Lemma 39.
- In the case where the current subterm is an application, apply Lemma 36.
- In the case where the current subterm is an abstraction and the stack is empty, apply Lemma 37.

- In the case where the current subterm is an abstraction and the stack is not empty, apply Lemma 40.

$\square$

Theorem 44 *For any λ-term t, we have*

$$l_h((t,\emptyset),\epsilon) = \inf\{|\Pi| \;/\; \exists(\Gamma,\alpha) \in \Phi \times D \text{ s.t. } \Pi \in \Delta(t,(\Gamma,\alpha))\} \;.$$

PROOF. We distinguish between two cases.

- The λ-term t is not head normalizable : apply Theorem 33 (i) and Theorem 9.
- The λ-term t is head normalizable: apply Proposition 42 and Proposition 43.

$\square$

5.3 Principal typings and 1-typings

In the preceding subsection, we related $l_h(t)$ and the size of the derivations of t for any λ-term t. Now, we want to relate $l_\beta(t)$ and the size of the derivations of t. We will show that if the value of $l_\beta(t)$ is finite (i.e. t is normalizable), then it is the size of the least derivations of t with typings that satisfy a particular property and that, otherwise, there is no such derivation. In particular, in the finite case, it is the size of the derivations of t with 1-*typings* of the normal form of t. This notion of 1-*typing*, defined in Definition 46, is a generalization of the notion of *principal typing*.

The work of [Coppo *et al.* 1980] can be adapted in order to show that all normal λ-terms have a principal typing in System R if A is infinite. A typing (Γ,α) for a λ-term is a principal typing if all other typings for the same λ-term can be derived from (Γ,α) by some set of operations. Here, the operations are substitution (see Definition 27) and expansion (complicated to define). The difference with [Coppo *et al.* 1980] is the fact that we have (and we need) the notion of 0-expansion ([Coppo *et al.* 1980] has the notion of n-expansion only for $n \geq 1$). For any (Γ,α), $(\Gamma',\alpha') \in \Phi \times D$, $(\Gamma,\alpha) \to (\Gamma',\alpha')$ will denote the fact that there exists an integer n such that (Γ',α') is a n-expansion of (Γ,α). We denote by $\to^*$ the reflexive transitive closure of $\to$.

Definition 45 *Principal typing of normal λ-terms :*

$$\frac{}{x : [\gamma] \vdash_P x : \gamma}\ \gamma \in A$$

$$\frac{\Gamma, x : a \vdash_P t : \alpha}{\Gamma \vdash_P \lambda x.t : (a,\alpha)}$$

$$\frac{\Gamma_1 \vdash_P u_1 : \alpha_1 \qquad \ldots \qquad \Gamma_n \vdash_P u_n : \alpha_n}{\sum_{i=1}^{n} \Gamma_i + \{(x,[[\alpha_1]\ldots[\alpha_n]\gamma])\} \vdash_P (x)u_1 \ldots u_n : \gamma}\ (*)$$

$(*)$ *the atoms in Γ_j are disjoint from those in Γ_k if $j \neq k$ and $\gamma \in A$ does not appear in the Γ_i*

If $\Delta \vdash_P t : \beta$, then (Δ, β) is said to be a principal typing of t. We could show that whenever (Δ, β) is a principal typing of a normal λ-term t, then we have $\Gamma \vdash_R t : \alpha$ if, and only if, there exists (Γ', α') such that $(\Delta, \beta) \to^* (\Gamma', \alpha')$ and (Γ, α) can be obtained from (Γ', α') by a substitution (exactly as in [Coppo *et al.* 1980], except that we consider the 0-expansions too). But we do not need this result to prove the theorems of the following subsection; we mention it only to justify the terminology we use.

The reader acquainted with the concept of *experiment* on proof nets in linear logic could notice that a principal typing of a normal λ-term is the same thing as the result of what [Tortora de Falco 2000] calls *an injective obsessional* 1*-experiment* of the proof net obtained by the translation of this λ-term mentioned in Subsection 3.2.

The notion of 1-typing is more general than the notion of principal typing. It is the result of an *obsessional* 1*-experiment*.

Definition 46 1*-typing of normal λ-terms :*

$$\frac{}{x : [\gamma] \vdash_1 x : \gamma}\, \gamma \in A$$

$$\frac{\Gamma, x : a \vdash_1 t : \alpha}{\Gamma \vdash_1 \lambda x.t : (a, \alpha)}$$

$$\frac{\Gamma_1 \vdash_1 u_1 : \alpha_1 \qquad \ldots \qquad \Gamma_n \vdash_1 u_n : \alpha_n}{\sum_{i=1}^n \Gamma_i + \{(x, [[\alpha_1] \ldots [\alpha_n]\gamma])\} \vdash_1 (x)u_1 \ldots u_n : \gamma}$$

Note that if t is a normalizable λ-term and (Γ, α) is a 1-typing of its normal form, then (Γ, α) has the following property: $[]$ has no positive occurrences in α and no negative occurrences in Γ.

Lemma 47 *Let (x, e) be a closure and let $\Gamma \in \Phi$ such that $[]$ has only positive occurrences in Γ. Assume that there exists a derivation of $\Gamma \vdash (x, e) : b_1 \ldots b_q \alpha$, with $x \notin dom(e)$, then for any $k \in \{1, \ldots, q\}$, we have $b_k \neq []$.*

Proof. Let Π be such a derivation. Set $e = \{(x_1, c_1), \ldots, (x_m, c_m)\}$.

We have $\Pi = (\Pi_0, \{(x_1, \Pi_1), \ldots, (x_m, \Pi_m)\})$, where

(i) Π_0 is a derivation of $\Gamma_0, x_1 : a_1, \ldots, x_m : a_m \vdash_R x : b_1 \ldots b_q \alpha$,

(ii) for $j \in \{1, \ldots, m\}$, Π_i is a derivation of $\Gamma_j \vdash c_j : a_j$

(iii) and $\Gamma = \sum_{j=0}^m \Gamma_j$.

By (i), since $x \notin \mathrm{dom}(e)$, $\Gamma_0(x) = [b_1 \ldots b_q \alpha]$. Hence, by (iii), if there existed $k \in \{1, \ldots, q\}$ such that $b_k = []$, then there would be a negative occurrence of $[]$ in Γ. $\Box$

Proposition 48 *Let t be a normalizable λ-term. If Π is a derivation of $\Gamma \vdash_R t : \alpha$ and (Γ, α) is such that $[]$ has only negative occurrences in α and only positive occurrences in Γ, then we have $l_\beta((t, \emptyset), \epsilon) \leq |\Pi|$.*

Proof. By Theorem 12, we can prove, by induction on $l_\beta(s)$, that for any $s = (c_0, (c_1, \ldots, c_q)) \in \mathbb{S}$ such that $(\overline{c_0})\overline{c_1} \ldots \overline{c_q}$ is normalizable, for any $(\Gamma, \alpha) \in \Phi \times D$, if Π is a derivation of $\Gamma \vdash s : \alpha$ and (Γ, α) is such that $[]$ has only negative occurrences in α and only positive occurrences in Γ, then we have $l_\beta(s) \leq |\Pi|$.

In the case where $s = ((x, e), (c_1, \ldots, c_q))$ and $x \notin \mathrm{dom}(e)$, we apply Lemma 47. $\Box$

Proposition 49 *Assume that t is a normalizable λ-term and that (Γ, α) is a 1-typing of its normal form. Then there exists a derivation Π of $\Gamma \vdash_R t : \alpha$ such that $l_\beta((t, \emptyset), \epsilon) = |\Pi|$.*

PROOF. By Theorem 12, we can prove, by induction on $l_\beta(s)$, that for any $s \in \mathbb{S}$ such that $\overline{s}$ is normalizable and for any 1-typing (Γ, α) of the normal form of $\overline{s}$, there exists a derivation Π of $\Gamma \vdash s : \alpha$ such that $l_\beta(s) = |\Pi|$.

The base case is trivial, because we never have $l_\beta(s) = 0$. The inductive step is divided into five cases:

- In the case where $s = ((x, e), (c_1, \ldots, c_q))$ and $x \notin \mathrm{dom}(e)$, (Γ, α) is a 1-typing of $(x)t_1 \ldots t_q$, where $t_1, \ldots, t_q$ are the respective normal forms of $\overline{c_1}, \ldots, \overline{c_q}$, hence there exist $\Gamma_1, \ldots, \Gamma_q$, $\alpha_1, \ldots, \alpha_q$ such that
  - $\Gamma = \sum_{k=1}^{q} \Gamma_k + \{(x, [[\alpha_1] \ldots [\alpha_q]\alpha])\}$
  - and $(\Gamma_1, \alpha_1), \ldots, (\Gamma_q, \alpha_q)$ are 1-typings of $t_1, \ldots, t_q$ respectively.

  By induction hypothesis, there exist q derivations $\Pi_1, \ldots, \Pi_q$ of $\Gamma_1 \vdash_R t_1 : \alpha_1, \ldots, \Gamma_q \vdash_R t_q : \alpha_q$ respectively. We denote by $x_1, \ldots, x_m$ the elements of $\mathrm{dom}(e)$. We denote by Π_0 the derivation of

  $$x : [[\alpha_1] \ldots [\alpha_q]\alpha] \vdash_R x : \alpha \ .$$

  Set $\Pi = ((\Pi_0, \{(x_1, \epsilon), \ldots, (x_m, \epsilon)\}), (\Pi_1, \ldots, \Pi_q))$: it is a derivation of $\Gamma \vdash_R \overline{s} : \alpha$ and we have

  $$\begin{array}{rcl} l_\beta(s) & = & \displaystyle\sum_{k=1}^{q} l_\beta(c_k) + 1 \\ & = & \displaystyle\sum_{k=1}^{q} |\Pi_k| + 1 \\ & & \text{(by induction hypothesis)} \\ & = & |\Pi_0| + \displaystyle\sum_{k=1}^{q} |\Pi_k| \\ & = & |((\Pi_0, \{(x_1, \epsilon), \ldots, (x_m, \epsilon)\}), (\Pi_1, \ldots, \Pi_q))| \\ & = & |\Pi| \ . \end{array}$$

- In the case where $s = ((x_{j_0}, \{(x_1, c'_1), \ldots, (x_m, c'_m)\}), (c_1, \ldots, c_q))$ with $1 \leq j_0 \leq m$, by induction hypothesis, there exists a derivation Π' of $\Gamma \vdash (c'_{j_0}, (c_1, \ldots, c_q)) : \alpha$ such that
  - t is the normal form of $(\overline{c'_{j_0}})\overline{c_1} \ldots \overline{c_q}$;
  - (Γ, α) is a 1-typing of t ;
  - and $l_\beta(c'_{j_0}, (c_1, \ldots, c_q)) = |\Pi'|$.

  By Lemma 39, there exists a derivation Π of $\Gamma \vdash s : \alpha$ such that $|\Pi| = |\Pi'| + 1$. We have

  $$\begin{array}{rcl} l_\beta(s) & = & l_\beta((c'_{j_0}, (c_1, \ldots, c_q)) + 1 \\ & = & |\Pi'| + 1 \\ & & \text{(by induction hypothesis)} \\ & = & |\Pi| \ . \end{array}$$

- In the case where $s = (((v)u, e), (c_1, \ldots, c_q))$, apply Lemma 36.
- In the case where $s = ((\lambda x.u, e), \epsilon)$, apply Lemma 37.
- In the case where $s = ((\lambda x.u, e), \pi)$ and $\pi \neq \epsilon$, apply Lemma 40.

$\square$

Theorem 50 *For any λ-term t, we have*

$$l_\beta((t, \emptyset), \epsilon) = \inf \left\{ |\Pi| \;/\; \begin{array}{c} \exists (\Gamma, \alpha) \text{ s.t. } \Pi \in \Delta(t, (\Gamma, \alpha)), \\ [] \textit{ has no positive occurences in } \alpha \\ \textit{and no negative occurrences in } \Gamma \end{array} \right\} .$$

PROOF. We distinguish between two cases.

- The λ-term t is not normalizable : apply Theorem 33 (ii) and Theorem 12.
- The λ-term t is normalizable : apply Proposition 48 and Proposition 49.

$\square$

5.4 Relating semantics and execution time

In this subsection, we prove the first truly semantic measure of execution time of this paper by bounding (by purely semantic means, i.e. without considering derivations) the number of steps of the computation of the principal head normal form (Theorem 55).

We define the size $|\delta|$ of any type δ and of any finite multiset δ of types, using an auxiliary function $\overline{s}$.

Definition 51 *For any $\delta \in D \cup \mathcal{M}_f(D)$, we define, by induction on δ, $|\delta|$ and $\overline{s}(\delta)$:*

- *if $\delta \in A$, then $|\delta| = 1$ and $\overline{s}(\delta) = 0$;*
- *if $\delta = [\alpha_1, \ldots, \alpha_n]$, then $|\delta| = \sum_{i=1}^n |\alpha_i|$ and $\overline{s}(\delta) = \sum_{i=1}^n \overline{s}(\alpha_i)$;*
- *if $\delta = (a, \alpha)$, then $|\delta| = \overline{s}(a) + |\alpha| + 1$ and $\overline{s}(\delta) = |a| + \overline{s}(\alpha) + 1$.*

Notice that for any $\alpha \in D$, the size $|\alpha|$ of α is the sum of the number of positive occurrences of atoms in α and of the number of commas separating a multiset of types and a type.

Example 52 *Let $\gamma \in A$. Set $\alpha = ([\gamma], \gamma)$ and $a = [\underbrace{\alpha, \ldots, \alpha}_{n \text{ times}}]$. We have $|(a, \alpha)| = 2n + 3$.*

Lemma 53 *For any λ-term u, if there exists a derivation Π of $x_1 : a_1, \ldots, x_m : a_m \vdash_R u : \alpha$, then $|a_1 \ldots a_m \alpha| = \overline{s}(a_1 \ldots a_m \alpha)$.*

PROOF. By induction on Π. $\square$

Lemma 54 *Let v be a normal λ-term and let Π be a derivation of $x_1 : a_1, \ldots, x_m : a_m \vdash_R v : \alpha$. Then we have $|\Pi| \leq |a_1 \ldots a_m \alpha|$.*

PROOF. By induction on v. □

Theorem 55 *Let v and u be two closed normal λ-terms. Assume $(a, \alpha) \in [\![v]\!]$ and $\mathsf{Supp}(a) \subseteq [\![u]\!]$.*

(i) We have

$$l_h(((v)u, \emptyset), \epsilon) \leq 2|a| + |\alpha| + 2 \ .$$

(ii) If, moreover, $[]$ has no positive occurrences in α, then we have

$$l_\beta(((v)u, \emptyset), \epsilon) \leq 2|a| + |\alpha| + 2 \ .$$

PROOF. Set $a = [\alpha_1, \ldots, \alpha_n]$. There exist a derivation Π_0 of $\vdash_R v : (a, \alpha)$ and n derivations $\Pi_1, \ldots, \Pi_n$ of $\vdash_R u : \alpha_1, \ \ldots, \vdash_R u : \alpha_n$ respectively. Hence there exists a derivation Π of $\vdash_R (v)u : \alpha$ such that $|\Pi| = \sum_{i=0}^{n} |\Pi_i| + 1$.

(i) We have

$$\begin{aligned} l_h(((v)u, \emptyset), \epsilon) &\leq \sum_{i=0}^{n} |\Pi_i| + 1 \\ &\quad \text{(by Proposition 42)} \\ &\leq |(a, \alpha)| + \sum_{i=1}^{n} |\alpha_i| + 1 \\ &\quad \text{(by Lemma 54)} \\ &= \sum_{i=1}^{n} \overline{s}(\alpha_i) + |\alpha| + 1 + |a| + 1 \\ &= \sum_{i=1}^{n} |\alpha_i| + |\alpha| + 1 + |a| + 1 \\ &\quad \text{(by Lemma 53)} \\ &= 2|a| + |\alpha| + 2 \ . \end{aligned}$$

(ii) The only difference with the proof of (i) is that we apply Proposition 48 instead of Proposition 42.

□

5.5 The exact number of steps

This subsection is devoted to giving the exact number of steps by purely semantic means. For arbitrary points $(a, \alpha) \in [\![v]\!]$ such that $\mathsf{Supp}(a) \subseteq [\![u]\!]$, it is clearly impossible to obtain an equality in Theorem 55, because there exist such points with different sizes.

The only equalities we have by now are Theorem 44 and Theorem 50, which use the size of the derivations. A first idea is then to look for points $(a, \alpha) \in [\![v]\!]$ such that

$\mathsf{Supp}(a) \subseteq [\![u]\!]$ with $|(a,\alpha)|$ equal to the sizes of the derivations used in these theorems. But there are cases in which such points do not exist.

A more subtle way out is nevertheless possible, and here is where the notions of equivalence between derivations and of substitution defined in Subsection 3.3 come into the picture. More precisely, using the notion of substitution, Proposition 58 (the only place where we use the non-finiteness of the set A of atoms through Fact 56 and Lemma 57) shows how to find, for any $\beta \in [\![t]\!]$, an element $\alpha \in [\![t]\!]$ such that $|\alpha| = \min\{|\Pi| \,/\, \Pi \in \Delta(t,\beta)\}$.

We remind that $A = D \setminus (\mathcal{M}_f(D) \times D)$. The equivalence relation $\sim$ has been defined in Definition 26 and the notion of substitution has been defined in Definition 27. We recall that we denote by $\mathcal{S}$ the set of substitutions.

Fact 56 *Let v be a normal λ-term and let Π be a derivation of $x_1 : b_1, \ldots, x_m : b_m \vdash_R v : \beta$. There exist $a_1, \ldots, a_m, \alpha$ and a derivation Π' of $x_1 : a_1, \ldots, x_m : a_m \vdash_R v : \alpha$ such that $\Pi' \sim \Pi$ and $|\Pi'| + m = |a_1 \ldots a_m \alpha|$. If, moreover, A is infinite, then we can choose Π' in such a way that there exists a substitution σ such that $\overline{\sigma}(a_1) = b_1, \ldots, \overline{\sigma}(a_m) = b_m$ and $\sigma(\alpha) = \beta$.*

Proof. By induction on v. □

In the case where A is infinite, the derivation Π' of the lemma is what [Coppo *et al.* 1980] calls a *ground deduction for v*.

Lemma 57 *Assume A is infinite. Let t be a closed normal λ-term, let $\beta \in D$ and let $\Pi \in \Delta(t,\beta)$. Then we have*

$$|\Pi| = \min\{|\alpha| \,/\, \alpha \in D \textit{ s.t. } \exists \Pi' \in \Delta(t,\alpha),\ \sigma \in \mathcal{S} \textit{ s.t. } (\Pi' \sim \Pi \textit{ and } \sigma(\alpha) = \beta)\}\ .$$

Proof. Apply Lemma 54 and Fact 56. □

Proposition 58 *Assume A is infinite. Let t be a closed normal λ-term and let $\beta \in [\![t]\!]$. We have* $\min\{|\Pi| \,/\, \Pi \in \Delta(t,\beta)\} = \min\{|\alpha| \,/\, \alpha \in [\![t]\!]$ *s.t.* $\exists \sigma \in \mathcal{S}$ *s.t.* $\sigma(\alpha) = \beta\}$.

Proof. Set

$$m = \min\{|\Pi| \,/\, \Pi \in \Delta(t,\beta)\}$$

and

$$n = \min\{|\alpha| \,/\, \alpha \in [\![t]\!] \text{ s.t. } \exists \sigma \in \mathcal{S} \text{ s.t. } \sigma(\alpha) = \beta\}\ .$$

First, we prove that $m \leq n$. Let $\alpha \in [\![t]\!]$ such that $\exists \sigma \in \mathcal{S}$ s.t. $\sigma(\alpha) = \beta$. By Theorem 20, $\Delta(t,\alpha) \neq \emptyset$: let $\Pi' \in \Delta(t,\alpha)$. By Proposition 28, there exists $\Pi \in \Delta(t,\beta)$ such that $\Pi \sim \Pi'$. By Lemma 54, we have $|\Pi'| \leq |\alpha|$. Hence we obtain $m \leq |\Pi| = |\Pi'| \leq |\alpha|$.

Now, we prove the inequality $n \leq m$. Let $\Pi \in \Delta(t,\beta)$.

$$\begin{array}{rcl} n & = & \min\{|\alpha| \,/\, \alpha \in D \text{ s.t. } \exists \Pi' \in \Delta(t,\alpha),\ \sigma \in \mathcal{S} \text{ s.t. } \sigma(\alpha) = \beta\} \\ & & \text{(by Theorem 20)} \\ & \leq & \min\{|\alpha| \,/\, \alpha \in D \text{ s.t. } \exists \Pi' \in \Delta(t,\alpha),\ \sigma \in \mathcal{S} \text{ s.t. } (\Pi' \sim \Pi \text{ and } \sigma(\alpha) = \beta)\} \\ & = & |\Pi| \\ & & \text{(by Lemma 57).} \end{array}$$

□

The point of Theorem 59 is that the number of steps of the computation of the (principal head) normal form of $(v)u$, where v and u are two closed normal λ-terms, can be determined from $[\![v]\!]$ and $[\![u]\!]$.

Theorem 59 *Assume A is infinite. Let v and u be two closed normal λ-terms.*

(i) We have

$$l_h(((v)u,\emptyset),\epsilon) = \inf \left\{ |(a,\alpha)| + |a'| + 1 \;/\; \begin{array}{l} (a,\alpha) \in [\![v]\!], a' \in \mathcal{M}_f(D) \textit{ s.t.} \\ \mathsf{Supp}(a') \subseteq [\![u]\!] \\ \textit{and } \exists \sigma \in \mathcal{S} \textit{ s.t. } \overline{\sigma}(a) = \overline{\sigma}(a') \end{array} \right\} .$$

(ii) We have

$$l_\beta(((v)u,\emptyset),\epsilon) = \inf \left\{ |(a,\alpha)| + |a'| + 1 \;/\; \begin{array}{l} (a,\alpha) \in [\![v]\!], a' \in \mathcal{M}_f(D) \textit{ s.t.} \\ \mathsf{Supp}(a') \subseteq [\![u]\!] \textit{ and} \\ \exists \sigma \in \mathcal{S} \textit{ s.t. } \overline{\sigma}(a) = \overline{\sigma}(a') \textit{ and} \\ [] \textit{ has no positive occurences in } \sigma(\alpha) \end{array} \right\} .$$

Proof.

(i) We distinguish between two cases.

- If $\Delta((v)u) = \emptyset$, then Theorem 44 shows that $l_h(((v)u,\emptyset),\epsilon) = \infty$ and Theorem 20 and Proposition 28 show that

$$\left\{ |(a,\alpha)| + |a'| + 1 \;/\; \begin{array}{l} (a,\alpha) \in [\![v]\!], a' \in \mathcal{M}_f(D) \text{ s.t.} \\ \mathsf{Supp}(a') \subseteq [\![u]\!] \\ \text{and } \exists \sigma \in \mathcal{S} \text{ s.t. } \overline{\sigma}(a) = \overline{\sigma}(a') \end{array} \right\} = \emptyset .$$

- Else, we have

$$\begin{array}{rl} & l_h(((v)u,\emptyset),\epsilon) \\ = & \min \left\{ \sum_{i=0}^{n} |\Pi_i| + 1 \;/\; \begin{array}{l} \exists \beta, \beta_1, \ldots, \beta_n \in D \text{ s.t.} \\ \Pi_0 \in \Delta(v, ([\beta_1,\ldots,\beta_n],\beta)), \\ \Pi_1 \in \Delta(u,\beta_1), \ldots, \Pi_n \in \Delta(u,\beta_n) \end{array} \right\} \\ & \text{(by Theorem 44)} \\ = & \min \left\{ \begin{array}{c} |([\alpha_1,\ldots,\alpha_n],\alpha)| \\ +\sum_{i=1}^n |\alpha'_i| \\ +1 \end{array} \;/\; \begin{array}{l} \alpha,\alpha_1,\ldots,\alpha_n,\alpha'_1,\ldots,\alpha'_n \in D \text{ s.t.} \\ ([\alpha_1,\ldots,\alpha_n],\alpha) \in [\![v]\!],\ \alpha'_1,\ldots,\alpha'_n \in [\![u]\!] \\ \text{and there exist } \sigma_0,\sigma_1,\ldots,\sigma_n \in \mathcal{S} \text{ s.t.} \\ \overline{\sigma_0}([\alpha_1,\ldots,\alpha_n]) = [\sigma_1(\alpha'_1),\ldots,\sigma_n(\alpha'_n)] \end{array} \right\} \\ & \text{(by applying Proposition 58 } n+1 \text{ times)} \\ = & \min \left\{ \begin{array}{c} |([\alpha_1,\ldots,\alpha_n],\alpha)| \\ +\sum_{i=1}^n |\alpha'_i| \\ +1 \end{array} \;/\; \begin{array}{l} \alpha,\alpha_1,\ldots,\alpha_n,\alpha'_1,\ldots,\alpha'_n \in D \text{ s.t.} \\ ([\alpha_1,\ldots,\alpha_n],\alpha) \in [\![v]\!],\ \alpha'_1,\ldots,\alpha'_n \in [\![u]\!] \\ \text{and there exists } \sigma \in \mathcal{S} \text{ s.t.} \\ \overline{\sigma}([\alpha_1,\ldots,\alpha_n]) = \overline{\sigma}([\alpha'_1,\ldots,\alpha'_n]) \end{array} \right\} \\ & \text{(the atoms in } \alpha'_1,\ldots,\alpha'_n \text{ can be assumed distinct} \\ & \text{and distinct of those in } \alpha_1,\ldots,\alpha_n). \end{array}$$

(ii) We distinguish between two cases.

- If $\left\{ \Pi \;/\; \begin{array}{l} \exists \beta \in D \text{ s.t.} \\ \Pi \in \Delta((v)u, \beta) \text{ and} \\ [] \text{ has no positive occurrences in } \beta \end{array} \right\} = \emptyset$, then Theorem 50 shows that $l_\beta(((v)u, \emptyset), \epsilon) = \infty$ and Theorem 20 and Proposition 28 show that

$$\left\{ |(a, \alpha)| + |a'| + 1 \;/\; \begin{array}{l} (a, \alpha) \in [\![v]\!], a' \in \mathcal{M}_f(D) \text{ s.t.} \\ \mathsf{Supp}(a') \subseteq [\![u]\!] \text{ and} \\ \exists \sigma \in \mathcal{S} \text{ s.t. } \overline{\sigma}(a) = \overline{\sigma}(a') \text{ and} \\ [] \text{ has no positive occurences in } \sigma(\alpha) \end{array} \right\} = \emptyset \;.$$

- Else, we have

$$\begin{array}{ll} & l_\beta(((v)u, \emptyset), \epsilon) \\ = & \min \left\{ \displaystyle\sum_{i=0}^{n} |\Pi_i| + 1 / \begin{array}{l} \exists \beta, \beta_1, \ldots, \beta_n \in D \text{ s.t.} \\ \Pi_0 \in \Delta(v, ([\beta_1, \ldots, \beta_n], \beta)), \\ \Pi_1 \in \Delta(u, \beta_1), \ldots, \Pi_n \in \Delta(u, \beta_n) \\ \text{and } [] \text{ has no positive occurrences in } \beta \end{array} \right\} \\ & \text{(by Theorem 50).} \\ = & \min \left\{ \begin{array}{c} |([\alpha_1, \ldots, \alpha_n], \alpha)| \\ + \sum_{i=1}^{n} |\alpha'_i| \\ +1 \end{array} / \begin{array}{l} \alpha, \alpha_1, \ldots, \alpha_n, \alpha'_1, \ldots, \alpha'_n \in D \text{ s.t.} \\ ([\alpha_1, \ldots, \alpha_n], \alpha) \in [\![v]\!], \; \alpha'_1, \ldots, \alpha'_n \in [\![u]\!] \\ \text{and } \exists \sigma_0, \sigma_1, \ldots, \sigma_n \in \mathcal{S} \text{ s.t.} \\ \overline{\sigma_0}([\alpha_1, \ldots, \alpha_n]) = [\sigma_1(\alpha'_1), \ldots, \sigma_n(\alpha'_n)] \\ \text{and } [] \text{ has no positive occurrences in } \sigma_0(\alpha) \end{array} \right\} \\ & \text{(by applying Proposition 58 } n+1 \text{ times)} \\ = & \min \left\{ \begin{array}{c} |([\alpha_1, \ldots, \alpha_n], \alpha)| \\ + \sum_{i=1}^{n} |\alpha'_i| \\ +1 \end{array} / \begin{array}{l} \alpha, \alpha_1, \ldots, \alpha_n, \alpha'_1, \ldots, \alpha'_n \in D \text{ s.t.} \\ ([\alpha_1, \ldots, \alpha_n], \alpha) \in [\![v]\!], \; \alpha'_1, \ldots, \alpha'_n \in [\![u]\!] \\ \text{and } \exists \sigma \in \mathcal{S} \text{ s.t.} \\ \overline{\sigma}([\alpha_1, \ldots, \alpha_n]) = \overline{\sigma}([\alpha'_1, \ldots, \alpha'_n]) \\ \text{and } [] \text{ has no positive occurrences in } \sigma(\alpha) \end{array} \right\} \\ & \text{(the atoms in } \alpha'_1, \ldots, \alpha'_n \text{ can be assumed distinct} \\ & \text{and distinct of those in } \alpha_1, \ldots, \alpha_n). \end{array}$$

□

Example 60 *Set $v = \lambda x.(x)x$ and $u = \lambda y.y$. Let $\gamma_0, \gamma_1 \in A$. Set*

- $\alpha = \gamma_0$;
- $a = [\gamma_0, ([\gamma_0], \gamma_0)]$;
- $a' = [([\gamma_1], \gamma_1), ([\gamma_2], \gamma_2)]$.

Let σ be a substitution such that $\sigma(\gamma_0) = ([\gamma_0], \gamma_0)$, $\sigma(\gamma_1) = \gamma_0$ and $\sigma(\gamma_2) = \alpha$. We have

- $(a, \alpha) \in [\![v]\!]$;
- $\mathsf{Supp}(a') \subseteq [\![u]\!]$;
- $\overline{\sigma}(a) = \overline{\sigma}(a')$;

- $|(a,\alpha)| = 4$ *and* $|a'| = 4$.

By Example 7, we know that we have $l_h(((v)u, \emptyset), \epsilon) = 9$. *And we have* $|(a,\alpha)| + |a'| + 1 = 9$.

Note that, as the following example illutrates, the non-idempotency is crucial.

Example 61 *For any integer* $n \geq 1$, *set* $\overline{n} = \lambda f.\lambda x.\underbrace{(f)\dots(f)}_{n\ times}x$ *and* $I = \lambda y.y$. *Let* $\gamma \in A$. *Set* $\alpha = ([\gamma], \gamma)$ *and* $a = [\underbrace{\alpha, \dots, \alpha}_{n\ times}]$. *We have* $(a, \alpha) \in [\![\overline{n}]\!]$ *and* $\alpha \in [\![I]\!]$. *We have* $l_h(((\overline{n})I, \emptyset), \epsilon) = 4(n+1) = 2n+3+2n+1 = |(a,\alpha)| + |a| + 1$ *(see Example 52). But in System* $\mathcal{D}$ *(with* idempotent *types), any Church integer* $\overline{n}$, *for* $n \geq 1$, *has type* $((\gamma \rightarrow \gamma) \rightarrow (\gamma \rightarrow \gamma))$.

6 Conclusion

We believe that this work can be useful for implicit characterizations of complexity classes (in particular, the PTIME class, as in [Baillot and Terui (2004)]) by providing a semantic setting in which quantitative aspects can be studied, while taking some distance with the syntactic details.

Note that if this paper, a redacted version of [de Carvalho 2006], concerns the λ-calculus and Krivine's machine, we emphasized connections with proof nets of linear logic. Because of these connections, we conjectured in [de Carvalho 2007] that we could obtain some similar results relating on the one hand the length of cut-elimination of nets with a strategy that mimics this one of Krivine's machine and that extends a strategy defined in [Mascari and Pedicini 1994] for a fragment of linear logic, and on the other hand the size of the results of experiments. This work has been done in [de Carvalho, Pagani and Tortora de Falco 2008] by adapting our work for the λ-calculus.

Acknowledgements. This work is partially the result of discussions with Thomas Ehrhard: I warmly thank him. I also thank Patrick Baillot, Simona Ronchi della Rocca and Kazushige Terui too for stimulating discussions.

References

[Barendregt 1984] Barendregt, H.P. (1984) *The Lambda Calculus. Its Syntax and Semantics*, revised edition. North-Holland.

[Baillot and Terui (2004)] Baillot, P. and Terui, K. (2004) Light types for polynomial time computation in lambda-calculus. In *Proceedings of LICS 2004, IEEE Computer Society Press*, 266–275.

[Benton *et al.* 1994] Benton, P.N., Bierman, G.M., de Paiva, V.C.V. and Hyland, J.M.E. (1992) Term assignment for intuitionistic linear logic. Technical Report 262, *Computer Laboratory, University of Cambridge.*

[Bierman 1993] Bierman, G.M. (1993) On Intuitionistic Linear Logic. PhD thesis, *University of Cambridge.*

[Bierman 1995] Bierman, G. M. (1995) What is a categorical model of intuitionistic linear logic? In *Proceedings of Conference on Typed Lambda Calculi and Applications*, volume **902**. Springer-Verlag.

[Boudol *et al.* 1999] Boudol, G., Curien, P.-L. and Lavatelli, C. (1999) A semantics for lambda calculi with resources. *Math. Struct. in Comp. Science* **9** (4), 437–482.

[Bucciarelli and Ehrhard 2001] Bucciarelli, A. and Ehrhard, T. (2001) On phase semantics and denotational semantics : the exponentials. *Annals of Pure and Applied Logic* **109** 205–241.

[de Carvalho 2006] de Carvalho, D. (2006) Execution time of Lambda-Terms via Non-Uniform Semantics and Intersection Types. *Preprint IML.*

[de Carvalho 2007] de Carvalho, D. (2007) Sémantiques de la logique linéaire et temps de calcul. PhD thesis, *Université Aix-Marseille 2.*

[de Carvalho, Pagani and Tortora de Falco 2008] de Carvalho, D., Pagani, M. and Tortora de Falco, L. (2008) A Semantic Measure of the Execution Time in Linear Logic. **RR 6441**, INRIA.

[Coppo *et al.* 1980] Coppo, M., Dezani-Ciancaglini, M. and Venneri, B. (1980) Principal type schemes and λ-calculus semantics. In J. P. Seldin and J. R. Hindley (editors), *To H. B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism*, 535–560. Academic Press.

[Dezani-Ciancaglini *et al.*] Dezani-Ciancaglini, M., Honsell, F. and Motohama, Y. (2005) Compositional characterisations of λ-terms using intersection types. *Theoretical Computer Science* **340** (3), 459-496.

[Ehrhard and Regnier 2006] Ehrhard, T. and Regnier, L. (2006) Böhm Trees, Krivine's Machine and the Taylor Expansion of Lambda-Terms. In A. Beckmann, U. Berger, B. Löwe and J. V. Tucker (editors), *Logical Approaches to Computational Barriers, Second Conference on Computability in Europe, CiE 2006, Swansea, UK, June 30-July 5, 2006, Proceedings*, 186–197. Springer-Verlag.

[Girard 1986] Girard, J. Y. (1986) The system F of variable types, fifteen years later. *Theoretical Computer Science* **45** (2), 159–192.

[Girard 1987] Girard, J. Y. (1987) Linear Logic. *Theoretical Computer Science* **50**, 1–102.

[Guerrini 2004] Guerrini, S. (2004) Proof Nets and the λ-calculus. In T. Ehrhard, J.-Y. Girard, P. Ruet and P. Scott (editors), *Linear Logic in Computer Science*, 65–118, Cambridge University Press.

[Kfoury *et al.* 1999] Kfoury, K., Mairson, H. G., Turbak, F. A. and Wells, J. B. (1999) Relating Typability and Expressiveness in Finite-Rank Intersection Types Systems (Extended Abstract), *ICFP*, 90–101.

[Kfoury 2000] Kfoury, A. J. (2000) A linearization of the Lambda-calculus and consequences. *Journal of Logic and Computation* **10** (3), 411–436.

[Krivine 1990] Krivine, J. L. (1990) *Lambda-calcul types et modèles.* Masson.

[Krivine 2007] Krivine, J. L. (2007) A call-by-name lambda-calculus machine. *Higher Order and Symbolic Computation* **20**, 199–207.

[Lafont 1988] Lafont, Y. (1988) Logiques, catégories et machines. PhD thesis, *Université Paris 7.*

[Mac Lane 1998] Mac Lane, S. (1998) *Categories for the Working Mathematician.* Springer-Verlag.

[Mascari and Pedicini 1994] Mascari, G. F. and Pedicini, M. (1994) Head linear reduction and pure proof net extraction. *Theoretical Computer Science* **135** (1), 111–137.

[Neergaard and Mairson 2004] Neergaard, P. M. and Mairson, H. G. (2004) Types, potency, and idempotency: why nonlinearity and amnesia make a type system work. In *ICFP '04: Proceedings of the ninth ACM SIGPLAN international conference on Functional programming*, 138–149, ACM Press.

[Regnier 1992] Regnier, L. (1992) Lambda-calcul et réseaux. PhD thesis, *Université Paris 7.*

[Ronchi Della Rocca 1988] Ronchi Della Rocca, S. (1988) Principal Type Scheme and Unification for Intersection Type Discipline. *Theoretical Computer Science* **59**, 181–209.

[Seely 1989] Seely, R. (1989) Linear logic, $*$-autonomous categories and cofree coalgebras. *Contemporary Mathematics* **92**.

[Selinger 2002] Selinger, P. (2002) The Lambda Calculus is Algebraic. *Journal of Functional Programming* **12** (6), 549-566.

[Tortora de Falco 2000] Tortora de Falco, L. (2000) Réseaux, cohérence et expériences obsessionnelles. PhD thesis, *Université Paris 7.*

Centre de recherche INRIA Nancy – Grand Est
LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399